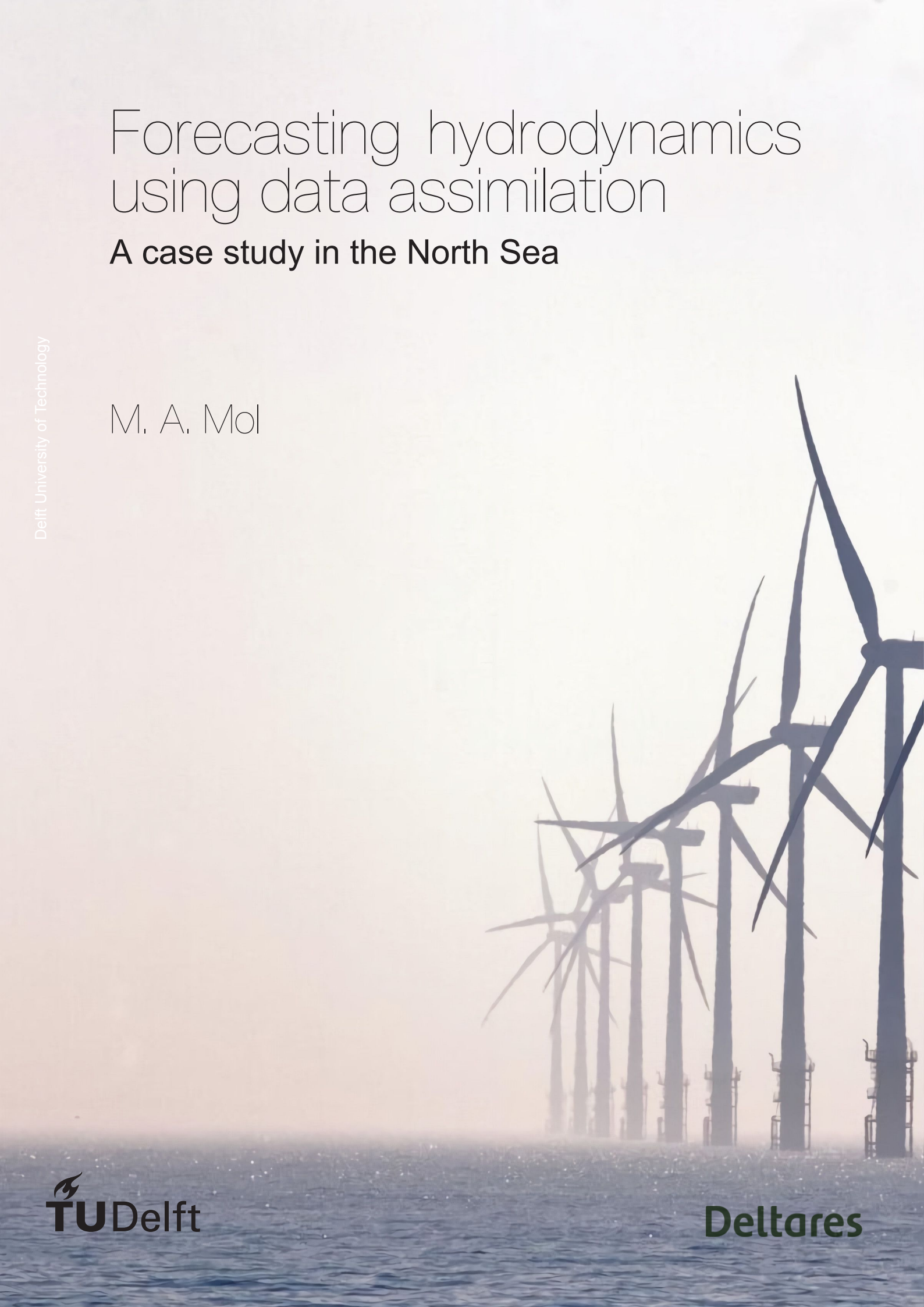
Forecasting hydrodynamics using data assimilation

A case study in the North Sea

M. A. Mol

Delft University of Technology



TU Delft

Deltares

Forecasting hydrodynamics using data assimilation

A case study in the North Sea

by

M. A. Mol

to obtain the degree of
Master of Science in Applied Mathematics
Specialisation Computational Science and Engineering

at the Delft University of Technology,
to be defended publicly on Tuesday August 31st, 2021 at 14:00 PM.

Student number:	4395913	
Future correspondence:	maaikemolmol@hotmail.com	
Project duration:	September 2, 2020 – August 31, 2021	
Thesis committee:	Prof. dr. ir. A. W. Heemink,	TU Delft Mathematical Physics, supervisor,
	Prof. dr. ir. M. B. van Gijzen,	TU Delft Numerical Analysis,
	Dr. G. Y. H. El Serafy,	Deltares, TU Delft, supervisor,
	MSc. E. de Korte,	Deltares, daily supervisor,

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Deltares

Abstract

The need for accurate estimation of hydrodynamic and water quality model variables arises from the UNITED project, which aims to create high-resolutional forecasting systems for monitoring the cultivation of seaweed and flat oysters and operating of the Belgian pilot of UNITED in the coastal area of the North Sea.
Accurate observations of physical variables are usually only known for small domains of the model, on the water surface. Therefore, data assimilation is applied on different hydrodynamic models to search for improved estimates of waterlevel, water velocity and temperature. The sequential data assimilation algorithm, the Ensemble Kalman filter is considered. The algorithm is a Monte-Carlo approximation of the Kalman filter, both using accurate observations assimilated into the model, where the model each time is shifted towards the observations when available, providing an optimal trade-off estimation between the model estimations and observations.
Two models are considered: a 1D model of the Western Scheldt estuary in Python and the 2D model with depth of the Western Scheldt estuary implemented with software packages Delft3D FM and OpenDA. Model estimates and estimates of model with assimilated observations are investigated in various experiments to look for improved predictions, using twin experiments to generate artificial observations over the whole domain. The various experiments contained the search for the effect of assimilation location on the estimates, of the effect of assimilation of different physical parameters, the effect of the observational error covariance and the effect of water surface assimilation on the estimation within the water column.
In most cases, the Ensemble Kalman filter improved the estimate of waterlevel and water velocity with varying results up to a decrease in rmse of a factor four, while the assimilation of temperature gave a worse prediction. Furthermore, assimilation of waterlevel gave the best improvement in estimate. For the investigated models, assimilation of observations near the boundary conditions with implied noise gave the most improved estimates. Furthermore, assimilation of water surface observations improved the estimate within the water column for water velocity with a factor of 2, while temperature assimilation did not show any improvement. The result for temperature may be due to a collapsing Ensemble Kalman filter due to small standard deviations and a not realistic model setup for temperature. Therefore, assimilation of only observations at the surface may be used to accurately improves water column estimations for water velocity. For temperature, more research is needed. Advised is an experiment with a different model setup with more realistic and longer spin-up time of temperature. The collapsing Ensemble Kalman filter can be prevented by initialization of new ensembles before the the collapse happens.
In further research, data assimilation can be applied on the DCSM model, a model involving the area of the Belgian pilot.

Keywords: Data assimilation, Ensemble Kalman Filter, hydrodynamical modelling, Delft3D FM, OpenDA

Preface

Dear reader,

In front of you lies the Master thesis “Forecasting hydrodynamics using data assimilation: A case study in the North Sea”, the end result based on the research I have been doing the last year in cooperation with Deltares. It celebrates the end of my student life at Delft University of Technology, a time I certainly enjoyed and where I gained so much knowledge. This goes beyond courses or study material. I believe the whole journey should be remembered and gained life lessons from, and I think looking back I accomplished that. So therefore, this will be my last few things I will say as a student of Delft University of Technology.

Conducting this research was not the easiest thing I have done in my student time. Doing this in the middle of a Corona pandemic working from home, not having a clear finished feeling at the end of the day, sometimes not having any progress for weeks and being your own worst critic, is maybe one of the most challenging things I have done ever. Nevertheless, in some time I will be most proud of the outcome. I couldn’t have done this all by myself and therefore I would like to thank a lot of people.

First of all, I would like to thank Deltares, for letting me into their company to conduct this research and to gain more knowledge in the field of ocean modelling and data science. Although unfortunately I wasn’t able to be present at the company, at least I have been able to meet you during beach walks or drinks outside of Deltares. I have learned a lot from the department meetings, intern meetings and meetings with supervisors. To that end, I would like to thank my supervisors. Arnold, I really learned a lot about data assimilation from our discussions, thank you for making me feel familiar with the content, in my opinion a very promising field of research. Ghada, thank you for your help during my thesis with getting understanding of the hydrodynamic model in Delft3D FM. Elias, thank you for keeping me motivated by asking all sorts of questions, for helping me with software problems and rephrasing problems during meetings. Furthermore, I want to thank Martin Verlaan, Erik Pelgrim, Stef Hummel and Julien Groeneboom for thinking along and helping with the implementation of the 2D model in Delft3D FM and OpenDA, even making adaptations in the software for the application in this thesis. I want to thank Martin van Gijzen for taking part in my thesis committee. I would like to thank the other interns from Deltares, for organizing the very interesting intern meetings inviting speakers from all over Deltares and especially Tobias and Marja for the occasional talks and studying together at the TU faculties and library.

Furthermore, I would like to thank some very important people on the background. My family and friends, Henk, Jelle and Marie, as well as Marijn, Ilse and Vera for unconditional support. Iris, Labs has doubled their income first half of 2021 because of us, thanks for proofreading my thesis and studying at the TU faculties. My roommates, Juul (Emma) and Karel for their understanding and stress relief. My far away friends from the PSI guesthouse, Jeremy, Cristina, Niek, Alex and Chrysi, I miss you a lot everyday. My futsal team from FC Tutor and the Dag Gilles club for all the fun we had. And all others I have met along the way but somehow didn’t end up here, you are in my mind as well, but I can’t name everyone. Lastly, I want to thank you for taking the time to read this thesis (or at least this preface) and hope you will gain insight and knowledge from it and hope you will find what your are searching for.

This Project has received funding from the European Union’s Horizon 2020 Research and innovation Programme under Grant Agreement no 862915.

M. A. Mol
Delft, August 2021

Contents

1

Introduction

The UNITED project [9] is a European project of Deltares [2], an independent institute for applied research in the field of water and subsurfaces, and 25 other contributing companies. The project was brought to life to provide evidence of the viability (economically, socially and environmentally) of multi-use of marine space (MU). Multi-use of marine space is intentional shared use of marine resources in close geographic proximity by two or more maritime activities. Figure 1.1 shows some of the maritime activities that can be combined: offshore windfarms, seaweed and/or mussel cultivation, flat oyster cultivation, floating solar energy by solar panels, fishery and tourism. MU allows for combined efficient operational planning of these maritime activities, and can increase both efficient usage of the ocean space and can decrease socio-economic costs and the impact on the environment [64].

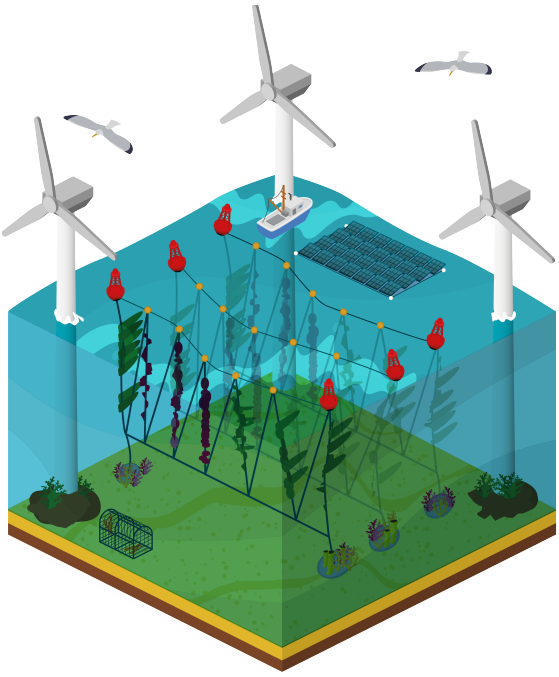


Figure 1.1: Schematic of multi-use of marine space, with maritime activies as offshore windfarms, seaweed and/or mussel cultivation, flat oyster cultivation, floating solar energy by solar panels, fishery and tourism. This is the ambition of multi-use of marine space what will be generally possible in 2030 [7, 48], by validation through the UNITED project. Figure retrieved from [84].

By setting up five offshore test locations or pilots in different coastal areas, with different combinations of marine activities, data is gathered and analyzed to obtain more information about MU, and to validate its use. The project started in 2020, and the pilot sites are currently in development. At the end of the project, UNITED aims to have created a validation of MU and raise the level of technology available

for MU activities within Europe [7]. Other companies and entrepreneurs will be able to start MU sites in the coastal regions more easily. The ambition for 2030 by one of the partners, North Sea Farmers, is to have 500 km^2 of multi-use farms within the Dutch North Sea [15, 84]. The list of UNITED pilots is given here and see Figure 1.2 for the locations:

1. Germany: Offshore windfarms, blue mussel and seaweed cultivation,
2. The Netherlands: Offshore windfarms, floating solar energy and seaweed cultivation,
3. Belgium: Offshore windfarms, flat oyster aquaculture and restoration and seaweed cultivation,
4. Denmark: Offshore windfarms and tourism,
5. Greece: Fish aquaculture and tourism.

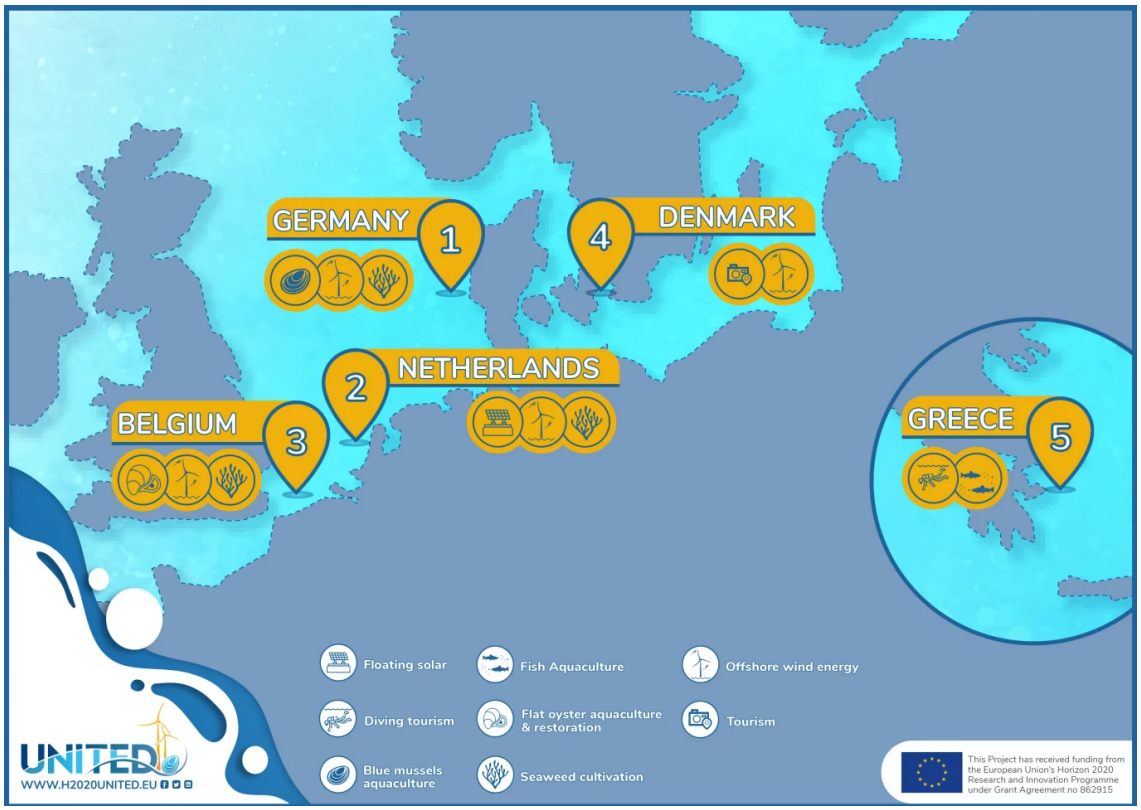


Figure 1.2: The five pilot sites of the UNITED project. In each pilot, multiple maritime activities are conducted. Figure retrieved from [9].

1.1. Belgian pilot

In this thesis, the Belgian pilot site is taken into consideration. The Belgian pilot aims to integrate flat oyster aquaculture and seaweed cultivation into existing windfarms, located approximately 38km off the Belgian coast [6]. This will be the first attempt to combine flat oyster aquaculture with existing windfarms. To that end, this section explains the three maritime activities to be conducted within the Belgian pilot: offshore windfarms, flat oyster aquaculture and restoration, and seaweed cultivation.
The windfarms in the Belgian coast are maintained by the company Parkwind [6]. There are 4 operational windfarms, with 771 MW of production capacity, stationed off the coast at Oostende, Belgium, to promote offshore wind energy development. This is roughly the amount to supply 820,000 EU households with electricity for one year [6].
The European flat oyster, also known as Ostrea edulis, has a large history within the North Sea. Since the Roman times, flat oysters were collected from the sea to grow in grow-out ponds, ponds where young fish are stocked and grown for consumption [8]. In the 18th and 19th century, flat oysters were

a main source of nutrition within Europe, resulting from large oyster reefs. Mainly due to overfishing, the flat oyster disappeared from the European waters. Furthermore, wild spat is becoming rare, and therefore aquaculture of flat oysters is very difficult in most rearing areas [8].

The creation of oyster reefs will be combined with seaweed cultivation. Seaweed is one of the fastest growing biomass in the world. It has many applications: in food, fuel, cosmetics, building materials and many more [10]. The demand for seaweed is increasing drastically with an averaged growth rate of 8% over the past few years [32]. Right now, it is cultivated for the most part in Asia by hand and with low quality. Therefore, specialized industrial production of seaweed of high-quality has big potential.

Combining these three maritime activities could strengthen eachother. Since fishery is not allowed in windfarms, potential oyster reefs will not be harmed by fishing nets. Furthermore, the solid bases of windturbines could be potential attachment points for aquaculture devices for seaweed cultivation [16], and will create a more diverse and resilient ecosystem also found on e.g. ship wrecks [51].

1.2. Problem statement

There is a lot of work involved in setting up offshore MU platforms and it entails various types of risk [49, 75]. The North Sea is a fairly rough sea and the distance from the coast results in more intense water conditions than near shore. This complicates the construction and operations using vessels.

Furthermore, optimal growing conditions of flat oysters and of seaweed are very specific [8]. Flat oysters are typically found in coastal areas, estuaries and marine habitats and prefer waters of relatively high salinity, with clear or low turbidity [21]. Flat oysters show high growth rates when salinity is above 22.5 psu (practical salinity units) [21], with optimal salinities of $24-34$ psu [57] and show high growth rates when temperature ranges between $17.5-30^{\circ}$ C. Growth is limited by for example the presence of algae [21].

The growth of seaweed is affected by light availability, wave behavior, water currents, nutrient concentration, type of substratum, temperature, salinity and grazing pressure [74]. The seaweed will grow adequately when sea water temperatures are confined between $5-15^{\circ}$C and summer sea water temperatures do not exceed $18-20^{\circ}$C for more than a few days, when salinity measures stay between $30-35$ psu, and when water motion is relatively moderate, around $0.1-0.25$ ms^{-1} [47]. Hence, limiting factors to the growth of both flat oysters and seaweed will be e.g. rough waters due to storms, algae blooms, high turbidity, high temperature, diverging salinity and oil spreading [37].

Another important process taking place in the North Sea is stratification [77]. Stratification is the forming of layers of unequal temperature within the ocean, caused by warming up of the top layers by the atmosphere or direct sunlight. Main consequences of stratification are the disruption of mixing in between the layers, lower density, lower salinity and higher temperature in the top layer. Due to higher temperature, less oxygen will be dissolved in the water. Low oxygen levels cause sediments to release nutrients in the water [12], which will stimulate the grow of algae [12], which in turn limits the growth of flat oysters and seaweed. The Belgian pilot is located on the border between permanently mixed and intermittently stratified areas [77]. Therefore, the amount of stratification will not be perfectly known throughout the year, which impacts clear decision-making in growthplans.

To safely plan constructions and schedule and maintain growthplans based on the desired growing conditions of flat oysters and seaweed, offshore MU platforms need high-resolutional hydrodynamic and water quality forecasting systems. These forecasting systems display and predict the status of physical parameters of a marine space and therefore can warn the pilot sites for potential problems. To that end, a high-resolutional prediction tool of hydrodynamics (i.e. water wave and velocity prediction) and waterquality (i.e. temperature, suspended particulate matter, chlorophyll-a) of the coastal area involving the UNITED pilot is desired.

Hydrodynamic and water quality models, for example in software such as Delft3D FM [24], give insight into the behavior of the physical parameters mentioned. Although models correctly display the physics behind the processes involved, the predictions can deviate a lot from the real situation and have large errors, as shown in Figure 1.3, due to several factors such as erroneous initial and boundary conditions.

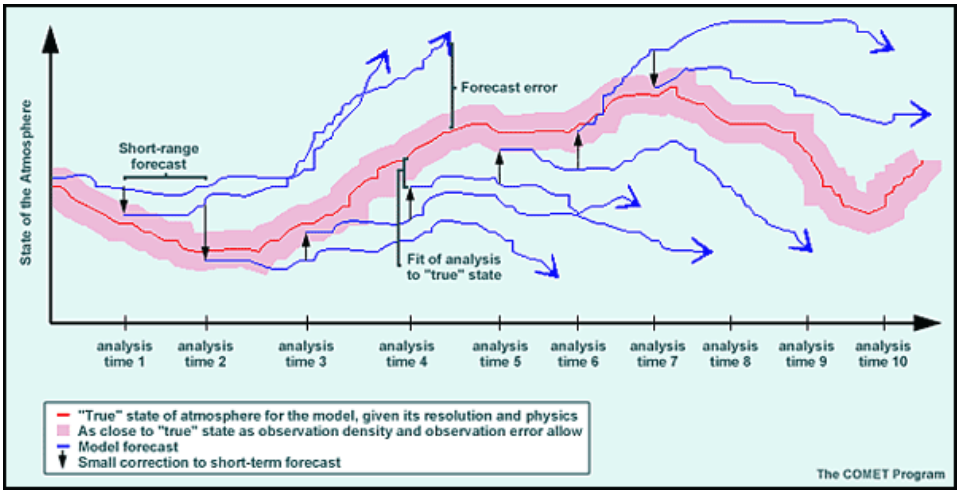


Figure 1.3: Forecast of a model simulating the atmosphere. The true state in red depicts the actual state the atmosphere is in. The model forecasts in blue are simulated with an initial state starting close to the true state. Since small errors in the initial conditions of the atmospheric model result in significantly different outcomes, the model forecasts diverge in time further from the true state. This is called the butterfly effect. Figure retrieved from [52].

Observational techniques used to gather measurements in the ocean are in situ and remote sensing, see Figure 1.4. In situ observations are measurements locally conducted at for example buoys measuring multiple physical variables directly within the water. Remote sensing is an observational technique indirectly determining the physical characteristics of an area by measuring its reflected and emitted radiation, making use of satellite or aircraft images. Observations are usually very precise, but scarce due to the limited amount of measuring devices and thus do not give information about the entire domain. In addition, observations are mostly only available on the water surface and not in the water column. Therefore, using only observations to predict on an entire 3D domain is not feasible.

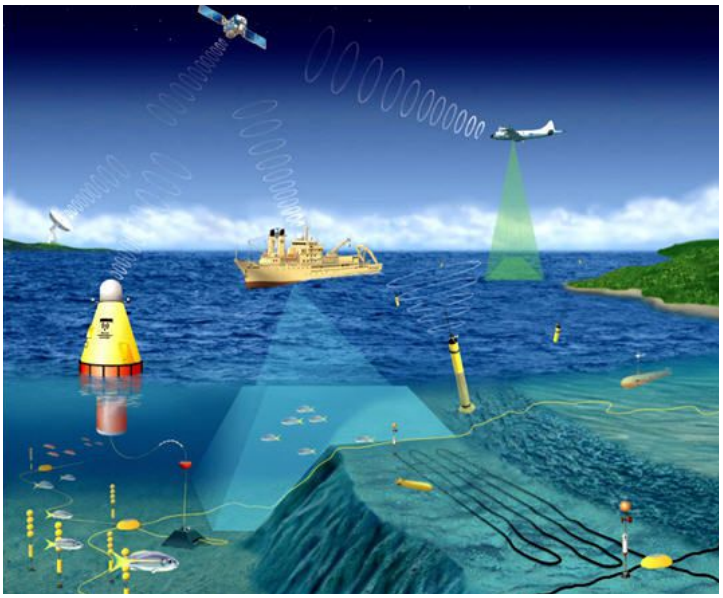


Figure 1.4: Various methods to obtain measurements within the ocean. Both in situ (local) measurements acquired via for example buoys, and remote sensing measurements via satellites, radars and aircrafts. Figure retrieved from [67].

Consequently, the combination of both numerical models and observations is the most beneficial for prediction of the ocean [37]. One of the most optimal techniques behind this specific concept is data assimilation. The usage of Artificial Intelligence (AI) as alternative is discouraged due to the scarcity of the measurements and since the necessary underlying physical models are disregarded within AI.

1.3. Related research in Data assimilation

In this section, related research to data assimilation and background information on this thesis is given. Generally, data assimilation is described as [11]: “The science of combining different sources of information to estimate possible states of a system as it evolves in time”. It makes use of optimization techniques, such that predictions do not deviate too far from assumed precise given measurements. Data assimilation is widely applied in forecasting systems with atmosphere, ocean and land surface predictions. There are two branches in data assimilation: variational and sequential methods. Variational methods, such as Three-dimensional Variational Data Analysis (3DVar) and Four-dimensional Variational Data Analysis (4DVar) will find an estimation that minimizes a cost function between observations and model. Sequential methods are e.g. the Kalman filter (KF), the Ensemble Kalman filter (EnKF), which stochastically estimate variables per timestep and update these estimates with observations. Generic implementation is possible with software packages as OpenDA [5]. In this thesis, the focus is put on sequential methods and mainly the Ensemble Kalman filter, which is a Monte-Carlo data assimilation algorithm for nonlinear systems, explained in detail in this thesis, as this filter is used frequently in similar applications.

Already in the seventeenth and eighteenth century, mathematicians and astronomers saw the use of observations while predicting orbits of comets and planets using the least squares criterion [53]. The first real data assimilation filter was implied by Kalman in 1960 [45]. The Ensemble Kalman Filter (EnKF) was first introduced by Evensen in 1994 [34]. Multiple studies have been conducted on the theory behind the Ensemble Kalman filter [17, 35, 36].

An extension on data assimilation with the goal to reduce spurious correlations is localization. It was first introduced by Houtekamer and Mitchell in 2001 [44], and further worked out in [40], where localization proved to reduce the total error on the prediction. Localization provide that observations will only have an effect in their close local neighborhood in space. It is said to be likely that the cost of localizing covariances will be significantly less than the cost of generating a large enough ensemble for the errors to be similar [40]. In [44], it is argued that applying localization effectively on only 10% of the globe is as effective as multiplying the number of ensemble members of the EnKF by a factor of 10.

1.3.1. History of forecasting systems

The field of forecasting systems started around 1922 with weather forecasting, when the first erroneous attempt to forecast the atmosphere was made by Richardson [65], producing a 6 hour forecast of the state of the atmosphere in two points in Europe. At that time, his dream for the future was that “Perhaps some day in the dim future it will be possible to advance the computations faster than the weather advances and at a cost less than the saving to mankind due to the information gained” [65], pointing at the very high computational costs of the method. This computational power was not available at that time. In 1950, the first successful 24 hour weather forecast was produced on a pioneering digital computer (ENIAC [83]) by a team of international scientists including Charney, Fjörtoft, and Von Neumann [20]. The calculations took 24 hours to complete, which is too long for a useful forecast, however the team suggested that the computational time could be reduced in the future. From 1954, operational numerical weather prediction began. Since then, computational power has improved drastically and numerical weather prediction has been researched thoroughly described in e.g. [59] and improved drastically, leading up to very accurate 7 day weather forecasts seen nowadays.

In 1954, forecasting of waves became relevant to scientists [14], when a part of The Netherlands flooded during the North Sea flood of 1953. Scientists had noticed that the atmosphere and ocean behave in a similar unpredictable way, and therefore could be coupled to eachother and same theory could be used [59]. In 1970, the first operational hemispheric wave model called Spectral Ocean Wave Model or SOWM was created. Around 1997, it was discussed that using data assimilation could optimize ocean forecasting models [37].

1.3.2. Applications Ensemble Kalman filter

The Ensemble Kalman filter is since its foundation in 1994 widely used in multiple fields of research were both model and observations are available, such as Oceanography (study of Earth’s oceans), Meteorology (study of the Earth’s atmosphere), Satellite Orbit Determination in space.

The Ensemble Kalman filter and Kalman filter are widely applied on various models in oceanography

where application of the filter improved the estimated, e.g. in storm surge forecasting models [46, 80, 87], in coastal models [33, 56], in state estimation in operational circulation models for the ocean and atmosphere as well as parameter estimation or history matching in reservoir simulation models [36].
Furthermore, multiple studies are conducted on the assimilation of temperature data with the EnKF on a shallow water model in the North Sea for 1D and 3D [60–63] by Ponsar et al.. In these studies, it is shown that the boundary of the stratified layers in the North Sea are not accurately displayed. Using temperature assimilation the boundary between the stratified layers is recovered. Furthermore, temperature assimilation in coastal models is investigated and this resulted in improved estimations. In [42], assimilation of sea surface temperature has been implemented on the MICOM model in the Indian Ocean.
Recent studies into high-resolutional forecasting systems are conducted [39, 71, 72], showing the feasibility of high-resolutional forecasting systems.
The DCSM model, a hydrodynamic model for the North Sea considered in this thesis, has been used for numerous studies [85, 86, 88]. There are also numerous studies where the OpenDA software is applied as data assimilation technique on various models, e.g. in [79] and in [13], where a data assimilation framework is made for 3D hydrodynamic lake models and applied on Lake Geneva.

1.4. Thesis approach

In this thesis, a start will be made to the realization of an operational high-resolutional hydrodynamic and water quality forecasting system for the area of the Belgian pilot of the UNITED project within the coastal area of the North Sea. In particular, this thesis puts the focus on the forecasting of the physical variables: waterlevel, water velocity and temperature. Hereby, the most generally applicable data assimilation algorithm, the Ensemble Kalman filter, is performed on different hydrodynamic models. This to investigate the accuracy and computational efficiency of the resulting forecasting system by uncertainty and sensitivity analyses, in comparison to only hydrodynamic model predictions. This thesis aims to give insight if the prediction in unobserved locations can be improved by using the Ensemble Kalman filter. Since observations are often only present at the water surface, this thesis aims to look at the improvement of the estimate within the water column as production of seaweed and flat oysters takes place within the water and the physical variables need to be known within the water column. The main research questions this thesis aims to answer are:

- How much does the implementation of the Ensemble Kalman filter on the hydrodynamic models improve the prediction of waterlevel, water velocity and temperature in comparison to the prediction of only the hydrodynamic model?
- Assimilation of which physical variable gives the greatest reduction in prediction errors?
- What is the influence of assimilation applied at different locations on the prediction?
- How much does the estimate through the water column in a 2D model with depth improve by assimilating only observations at the surface for e.g. water velocity or temperature?

To that end, the approach of this thesis will be the implementation and validation of the Ensemble Kalman filter onto coastal models and estuaries. An estuary is a river which on one side reaches the ocean. The models considered are:

- A linear 1D shallow water hydrodynamic model of the Western Scheldt estuary with Ensemble Kalman filter in Python,
- A nonlinear 2D hydrodynamic model of an estuary with Ensemble Kalman filter in Delft3D FM and OpenDA,
- The 3D Delft Continental Shelf Model (DCSM), a nonlinear model based on the North Sea, in Delft3D FM and OpenDA.

The 1D shallow water hydrodynamic model is a straightforward model that will allow the understanding of the settings of the Ensemble Kalman filter, the more complex 2D hydrodynamic model will introduce the implementation of the Ensemble Kalman filter in data assimilation software package OpenDA on a

model within Delft3D FM and will investigate the assimilation on the water surface. Lastly, the foundations of both of these models will be needed to understand the DCSM model and it will be described how to proceed with this model to arrive at a high-resolutional forecasting system. The assimilated models will be validated with twin experiments by generation of synthetic observations, a common approach to asses data assimilation. Experiments are conducted showing the accuracy of all considered models.

In this thesis, the mathematical notation of state-space representation is introduced in Chapter 2. Chapter 3 describes the theory of hydrodynamic models in one dimension and discusses the scientific software packages as Delft3D FM modelling hydrodynamic and water quality models used for the 2D and 3D models in this thesis. Furthermore, Chapter 4 will discuss data assimilation in detail, in particular the Kalman filter and Ensemble Kalman filter and will describe the software package OpenDA used for data assimilation. Chapter 5 describes the experimental setup of the experiments in detail, whereas Chapter 6 will focus on the 1D hydrodynamic model with Ensemble Kalman filter in Python and shows results of executed experiments. Chapter 7 focuses on the 2D hydrodynamic model implemented within Delft3D FM and OpenDA, Chapter 8 will focus on the 3D DCSM model and will describe further steps to be made with this model. Chapter 9 provides the conclusions of this thesis and will present some recommendations for future research.

2

State-space representation

In this chapter, some general mathematical notation is introduced needed in this thesis. State-space representation for model evolution (section 2.1), the stochastic nature of these models (section 2.2) and the true state these models are representing (section 2.3) are discussed.

2.1. State-space representation

Within mathematics, models, such as a hydrodynamic model, are a way to estimate certain real life processes. A mathematical way to describe the evolution of models over time is by state-space representation. State-space representation describes the evolution of a state vector $\mathbf{x}_t \in \mathbb{R}^n$, a vector containing all available n state variables of the underlying model at a specific time t, as:

$$\mathbf{x}_t = \left(x_t^1, \dots, x_t^n\right)^T, \tag{2.1}$$

with x_t^i for $i = 1, \dots, n$ the state variable i at time t. The state vector can consist of multiple different state variables, for example physical variables as waterlevel at a specific position, water velocity at specific positions, temperature and salinity. The model is usually denoted by a set of differential or difference equations. In these models, the time derivative of each state variable is expressed in terms of the state vector $\mathbf{x}_t$ and some system input $\mathbf{u}_t \in \mathbb{R}^r$, with r inputs, containing for example boundary conditions, are written as:

$$\begin{aligned} \dot{x}_t^1 &= M_t^1(\mathbf{x}_t, \mathbf{u}_t, t) \\ \dot{x}_t^2 &= M_t^2(\mathbf{x}_t, \mathbf{u}_t, t) \\ \vdots &= \quad \vdots \\ \dot{x}_t^n &= M_t^n(\mathbf{x}_t, \mathbf{u}_t, t), \end{aligned}$$

where $\dot{x}_t^i = \frac{\mathrm{d}x_t^i}{\mathrm{d}t}$ and the model operator M_t^i for state variable i at time t for $i = 1, \dots, n$. The model operator may be a general nonlinear, time-varying function. In vector notation, it is written as

$$\dot{\mathbf{x}}_t = \mathbf{M}_t(\mathbf{x}_t, \mathbf{u}_t, t). \tag{2.2}$$

When a set of inital conditions $\mathbf{x}_{t_0}$ and input $\mathbf{u}_t$ at each time t with boundary conditions, the state at each proceeding time is found by integration.
In this thesis the focus is put on sequential data assimilation, governed by discrete time algorithms. Therefore, only discrete time is considered. In discrete time $k \in \mathbb{Z}$ a general, nonlinear time-varying model evolves according to a first-order nonlinear difference equation as

$$\mathbf{x}_{k+1} = \mathbf{M}_{k+1}(\mathbf{x}_k, \mathbf{u}_k, k). \tag{2.3}$$

When initial conditions $\mathbf{x}_0$ and input $\mathbf{u}_k$ for all $k > 0$ is given, the state at each proceeding time is found by performing the model operator $\mathbf{M}_{k+1}$ and finding $\mathbf{x}_{k+1}$ in a sequential way.
For linear time-independent models, Equation 2.3 reduces to a matrix multiplication as

$$\mathbf{x}_{k+1} = \mathbf{M}_k\mathbf{x}_k + \mathbf{B}_k\mathbf{u}_k, \tag{2.4}$$

with $\mathbf{M}_k$ a $n \times n$ matrix containing the model specifics and $\mathbf{B}_k$ a $n \times r$ matrix containing the input specifics.

In addition to sequential model evolution, state-space representation includes observations $\boldsymbol{z}_k \in \mathbb{R}^m$ with m the amount of observations, of some of the state variables as

$$\boldsymbol{z}_k = \left(z_k^1, \ldots, z_k^m\right)^T. \tag{2.5}$$

Usually the amount of observations is much lower than the amount of state variables, so $m \ll n$. The observations are linked to the specific state variables with

$$\boldsymbol{z}_k = \mathbf{H}_k(\mathbf{x}_k, \mathbf{u}_k), \tag{2.6}$$

where $\mathbf{H}_k$ is the possibly nonlinear time-dependent operator projecting the model space onto the observation space. In this thesis, it is assumed $\boldsymbol{z}_k$ is only dependent on the state vector $\mathbf{x}_k$. For linear models, Equation 2.6 reduces to

$$\boldsymbol{z}_k = \mathbf{H}_k \mathbf{x}_k. \tag{2.7}$$

$\mathbf{H}_k$ is an $m \times n$ matrix, linking each observation z_k^i to a state variable. When all observations z_k^i are directly related to a state variable x_k^j, $\mathbf{H}_k$ is a sparse matrix containing only zeros and ones.
To summarize, the general state-space representation is given as,

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{M}_{k+1}(\mathbf{x}_k, \mathbf{u}_k, k) \\ \boldsymbol{z}_k = \mathbf{H}_k(\mathbf{x}_k, \mathbf{u}_k), \end{cases} \tag{2.8}$$

while the linear state-space representation is stated as

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{M}_k \mathbf{x}_k + \mathbf{B}_k \mathbf{u}_k \\ \boldsymbol{z}_k = \mathbf{H}_k \mathbf{x}_k. \end{cases} \tag{2.9}$$

2.2. Deterministic and stochastic models

In addition, multiple classes of models exist: deterministic and stochastic models. In deterministic models, every model run with the same model representation and same initial conditions will have the same estimate, and therefore no noise is present in the model. In Equation 2.8 and Equation 2.9, the deterministic case of the state-space representation is denoted. In stochastic models, however, the estimate in each separate model run changes due to added noise $\mathbf{w}_k$ in the system. This noise can enter in multiple ways: via initial conditions, boundary conditions, the model and observation errors. It is assumed all noise in this thesis is zero mean white noise, thus normally distributed with $\mathrm{E}(\mathbf{w}_k) = 0$ and $\mathrm{Cov}(\mathbf{w}_k) = \mathbf{Q}_k$ and thus $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k)$. In the case that $\mathbf{Q}_k$ is symmetric and positive definite, then $\mathbf{w}_k$ is Gaussian white noise. In this thesis, model noise as $\mathbf{w}_k$ and observation noise $\mathbf{v}_k$ are considered. The observation noise is distributed as $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$, where $\mathbf{R}_k = \mathrm{Cov}(\mathbf{v}_k)$. $\mathbf{w}_k$ and $\mathbf{v}_k$ are added to the state representation in Equation 2.8 and 2.9 as

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{M}_{k+1}(\mathbf{x}_k, \mathbf{u}_k, k) + \mathbf{w}_k \\ \boldsymbol{z}_k = \mathbf{H}_k(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{v}_k, \end{cases} \tag{2.10}$$

and for the linear case

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{M}_k \mathbf{x}_k + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k \\ \boldsymbol{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k. \end{cases} \tag{2.11}$$

Now both state-space representations in Equation 2.10 and 2.11 are stochastic of nature, meaning every estimate $\mathbf{x}_{k+1}$ in separate model runs is different.

2.3. True state vector

It is assumed models and thus state-space representations derive a state vector estimate $\mathbf{x}_k$ of a true or actual state vector, $\mathbf{x}_k^{tr}$ at time k. The true state vector is the state vector the system is actually in. This vector usually is not known as models are estimating this true state vector. However, to investigate

how well the derived estimates are compared to the true state, methods are needed to generate a true state, by using observations and model predictions. These are so called twin experiments, explained further in section 5.2. In this thesis, the optimal estimates of true states are investigated. Therefore, the error statistics between the true and estimated state need to be considered. The error between the true state $\mathbf{x}_k^{tr}$ and an estimated model state vector $\mathbf{x}_k$ is given as:

$$\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}_k^{tr}, \tag{2.12}$$

and furthermore the model error covariance is defined as:

$$\mathbf{P}_k = \mathrm{Cov}(\mathbf{x}_k^{tr} - \mathbf{x}_k) = \mathbb{E}\big((\mathbf{x}_k^{tr} - \mathbf{x}_k)(\mathbf{x}_k^{tr} - \mathbf{x}_k)^T\big). \tag{2.13}$$

The error between observations and the true state is given as:

$$\mathbf{e}_k^o = \mathbf{z}_k - \mathbf{H}_k \mathbf{x}_k^{tr}. \tag{2.14}$$

3

Hydrodynamic models

Hydrodynamics is the study of liquids in motion and therefore includes the study of coastal areas, as shallow waters and estuaries. The motion of water is described by a hydrodynamic model. Hydrodynamic models in general are nonlinear systems, described by nonlinear partial differential equations. Solutions of hydrodynamic models will give insight into the behavior of waterlevel and water velocity. In this chapter, the 1D linear case for shallow waters in estuaries is considered, since a shallow water model for estuaries is studied in this thesis. Furthermore the 2D and 3D nonlinear models are explained for the more complex models. The shallow water hydrodynamic model will be analytically derived in section 3.1, prosecuted with a description of boundary conditions for this model in section 3.2. The numerical discretization of the resulting model is described in section 3.3. The stochastic behavior of hydrodynamic models is described in section 3.4. Lastly, the nonlinear models and software packages for modelling hydrodynamic models and waterquality models are considered only computationally in section 3.5.

3.1. Linear 1D hydrodynamic model

Every hydrodynamic model stems from three main conservation laws: conservation of mass, conservation of momentum and conservation of energy. Conservation of mass states that for any closed system with no mass entering or leaving the system, the mass M of the system must remain constant over time t. Conservation of momentum states that for any closed system, in this case with fluids, the amount of momentum ρu with density ρ and velocity u, remains constant over time. Conservation of energy, also known as the First Law of Thermodynamics, states that for any closed system, the total energy remains constant over time. Energy in systems are present in the shape of kinetic energy, potential energy and heat and can be converted to other forms of energy within the system. Inflow and outflow of mass, momentum and energy define how the mass, momentum and energy of the system will change over time.

In this section, a linear 1D hydrodynamic model for shallow water estuaries is derived by using the laws of conservation of mass and conservation of momentum. To that end, assume a system of infinitesimal small element Δx of an 1D estuary of length L from x to $x + \Delta x$ of constant width b, flowing water with density ρ as in Figure 3.1. The waterlevel is denoted as $\eta(x,t)$ and the water velocity with $u(x,t)$. The river bed is possibly curved. Assume a time-dependent volume $V(x,t)$ in 1D of the system, defined as

$$V(x,t) = \eta(x,t)b\Delta x. \tag{3.1}$$

Assume an inflow (discharge) $Q(x,t)$ and outflow $Q(x+\Delta x,t)$ defined by:

$$Q(x,t) = u(x,t)\eta(x,t)b. \tag{3.2}$$

Assume no energy or mass leaves the system in any other way than via the left and right boundary. The mass $M(t)$ of the system at a time t is defined by the density ρ, assumed constant by assuming an incompressible fluid by the shallow water approximation, and volume $V(x,t)$ as,

$$M(x,t) = \rho V(x,t) = \rho\eta(x,t)b\Delta x. \tag{3.3}$$

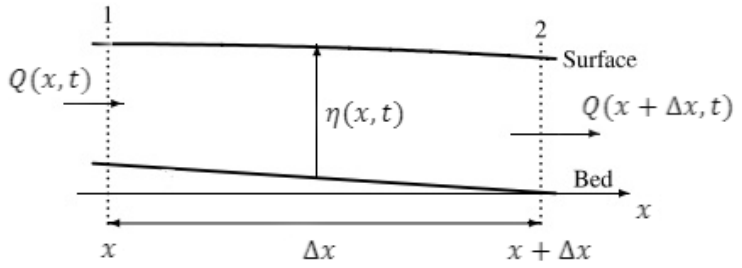


Figure 3.1: Longitudinal view of an infinitesimal small element of an 1D estuary with width Δx. The waterlevel is denoted by $\eta(x,t)$, the water velocity with $u(x,t)$. Water flows in and out of the system with an inflow $Q(x,t)$ and outflow $Q(x+\Delta x,t)$ respectively. Figure retrieved from [54] and adapted.

The mass inflow and outflow is defined by $\rho Q(x,t)$. The system must adhere to the law of conservation of mass. Due to the conservation of mass, mass cannot be created or destroyed. Therefore, the only way to increase or decrease the amount of mass is by in- or outflow of water. The change in mass is equal to the change in flow:

$$\frac{\mathrm{d}M}{\mathrm{d}t} = \rho Q(x,t) - \rho Q(x+\Delta x,t). \tag{3.4}$$

The change in mass is simplified by substituting M in Equation 3.3 into the left-hand side of Equation 3.4 and assuming density ρ is constant as:

$$\frac{\mathrm{d}M}{\mathrm{d}t} = \frac{\mathrm{d}\big(\rho\eta(x,t)b\Delta x\big)}{\mathrm{d}t} = \rho b\Delta x\frac{\partial\eta(x,t)}{\partial t}. \tag{3.5}$$

The change in flow is simplified by substituting Q in Equation 3.2 into the right-hand side of Equation 3.4:

$$\begin{aligned}\rho Q(x,t) - \rho Q(x+\Delta x,t) &= \rho u(x,t)\eta(x,t) - \rho u(x+\Delta x,t)\eta(x+\Delta x,t)\\ &= -\rho\frac{\partial(u\eta)}{\partial x}\Delta x.\end{aligned} \tag{3.6}$$

Substituting the result of Equation 3.5 and Equation 3.6 in Equation 3.4 gives

$$\frac{\partial\eta}{\partial t} = -\frac{\partial(u\eta)}{\partial x}. \tag{3.7}$$

This equation describes the change of waterlevel in time.

In a similar way described in for example [73], the law of conservation of momentum holds. Production of momentum in the system is generated by the pressure p acting on the system. By neglecting body forces such as gravity, molecular transport and shear stresses in the system but adding friction with $-\lambda u$, with friction coefficent λ, the resulting change of momentum is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = -\frac{\partial p}{\partial x}\frac{1}{\rho} - \lambda u. \tag{3.8}$$

This is a form of the Navier-Stokes equation in 1D, see e.g. [66], where body forces, shear stresses and molecular transport are neglected. By assuming $\eta \ll L$ in shallow waters, hydrostatic pressure follows [70] e.g. pressure from the water which increases by increasing depth as $p = \rho g\eta$, with g the gravitational acceleration, the pressure term in Equation 3.8 is simplified to

$$\frac{\partial p}{\partial x}\frac{1}{\rho} = g\frac{\partial\eta}{\partial x}, \tag{3.9}$$

which results into

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = -g\frac{\partial\eta}{\partial x} - \lambda u. \tag{3.10}$$

Therefore, the generalized 1D hydrodynamic model for shallow waters is expressed via Equation 3.7 and Equation 3.10 in the following way:

$$\begin{cases} \frac{\partial \eta}{\partial t} = -\frac{\partial (u\eta)}{\partial x}, \\ \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = -g\frac{\partial \eta}{\partial x} - \lambda u. \end{cases} \tag{3.11}$$

These are linearized by taking $\eta = D + h$, where D is the mean water level, assumed constant, and h is the height deviation from the mean waterlevel, and assuming h and u are very small. Therefore, products of two small variables, such as $u\frac{\partial h}{\partial x}$ are neglected and this results in

$$\frac{\partial (h+D)}{\partial t} + \frac{\partial \big(u(h+D)\big)}{\partial x} = 0 \implies \frac{\partial h}{\partial t} + D\frac{\partial u}{\partial x} = 0.$$

The second equation in Equation 3.11 is simplified by neglecting the convective acceleration $u\frac{\partial u}{\partial x}$ since it is a product of two small variables, to

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = -g\frac{\partial (h+D)}{\partial x} - \lambda u \implies \frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} + \lambda u = 0.$$

The resulting equations are known as the linearized shallow water equations:

$$\begin{cases} \frac{\partial h}{\partial t} + D\frac{\partial u}{\partial x} = 0 \\ \frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} + \lambda u = 0. \end{cases} \tag{3.12}$$

With the derivation of the 1D shallow water hydrodynamic model, the following is assumed for shallow waters:

- The water is incompressible, thus the density ρ is constant.
- $\eta \ll L$, resulting in hydrostatic pressure: $p = \rho g \eta$ [70].
- Product of the small variables h and u, such as $u\frac{\partial h}{\partial x}$ and $u\frac{\partial u}{\partial x}$ are neglected.
- Body forces, shear stresses and molecular transport is neglected [66].

3.2. Boundary conditions

To fully describe a hydrodynamic model, the set of differential equations needs boundary conditions. Both open and closed boundaries are possible. Open boundary conditions come from cases where water can flow outside of the boundary, which happens in e.g. oceans and cases where estuaries reach oceans. Closed boundaries are present when the water reaches the coast. These are represented by for example Neumann, Dirichlet and Riemann conditions, but also by discharge and outflow boundary conditions as described in [29].
Recall that an estuary is river channel which is on one side ending in the ocean. To that end, describing the estuary boundary conditions can be done by an open boundary in the sea at $x = 0$ (open boundary) and the other a closed end at $x = L$ (reflecting boundary), as respectively:

$$h(0,t) = h(t), u(L,t) = 0, \tag{3.13}$$

with L the end of the estuary. Therefore, the 1D shallow water equations in Equation 3.12 together with the set of boundary conditions in Equation 3.13 define the hydrodynamic model of an estuary, and the solutions contain the waterlevel $h(x,t)$ and velocity $u(x,t)$.

3.3. Numerical model 1D estuary

In order to approximate the solution to the hydrodynamic model, numerical models are needed. To that end, the shallow water hydrodynamic model in Equation 3.12 with boundary conditions for estuaries in Equation 3.13 is discretized in this section, where a new solution at timestep $k+1$ based on the

previous timestep k is derived. Assume a 1D grid with length L and grid size or resolution Δx and n grid points, see Figure 3.2.

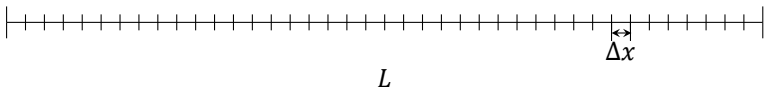


Figure 3.2: A 1D grid of length L with grid size or resolution Δx and n grid points.

The solutions of $h(x,t)$ and $u(x,t)$ are defined on this grid. To that end, the discretization of the solution in space will use a staggered grid notation, which indicates that waterlevel components will be stored on grid points and velocity components in between grid points. This discretization is used in favor of defining $h(x,t)$ and $u(x,t)$ on the same grid point (collocated discretization) since this is preferred for incompressible fluids, as collocated grids lead to unstabilities [41]. Therefore, the waterlevel and velocity at each locations are assembled in a state vector $\mathbf{x}_k$ at a time step k (see section 2.1), as the vector with all solutions at each location for a specific time k:

$$\mathbf{x}_k = \left(h_k^1, u_k^{1\frac{1}{2}}, h_k^2, u_k^{2\frac{1}{2}}, \dots, u_k^{n-\frac{1}{2}}, h_k^n, u_k^{n+\frac{1}{2}}\right)^T,$$

where h_k^i, u_k^i are the waterlevel and respectively velocity at x-position i at timestep k, see Figure 3.3.

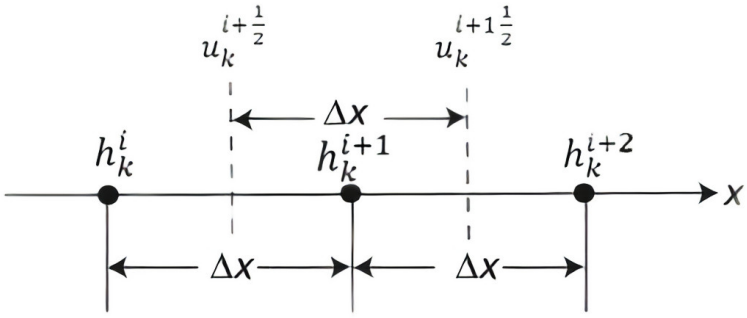


Figure 3.3: A staggered grid discretization in space, where waterlevel components are defined on grid points and velocity components in between grid points.

Assume time is discretized by forward difference scheme, with a timestep Δt between two times k and $k+1$. Then the time differentials in Equation 3.12 will be approximated by

$$\frac{\partial h}{\partial t} \approx \frac{h_{k+1}^i - h_k^i}{\Delta t} + \mathcal{O}(\Delta t); \quad \frac{\partial u}{\partial t} \approx \frac{u_{k+1}^{i+\frac{1}{2}} - u_k^{i+\frac{1}{2}}}{\Delta t} + \mathcal{O}(\Delta t). \tag{3.14}$$

The position differentials in Equation 3.12 will be approximated by:

$$\frac{\partial h}{\partial x} \approx \frac{h_{k+1}^{i+1} - h_{k+1}^i + h_k^{i+1} - h_k^i}{2\Delta x} + \mathcal{O}(\Delta x + \Delta t); \frac{\partial u}{\partial x} \approx \frac{u_{k+1}^{i+\frac{1}{2}} - u_{k+1}^{i-\frac{1}{2}} + u_k^{i+\frac{1}{2}} - u_k^{i-\frac{1}{2}}}{2\Delta x} + \mathcal{O}(\Delta x + \Delta t). \tag{3.15}$$

Then, combining Equation 3.12 and Equation 3.14 and Equation 3.15 results into the following discretization scheme:

$$\begin{cases} h_{k+1}^i + \frac{1}{2} D \frac{\Delta t}{\Delta x}\left(u_{k+1}^{i+\frac{1}{2}} - u_{k+1}^{i-\frac{1}{2}}\right) = h_k^i - \frac{1}{2} D \frac{\Delta t}{\Delta x}\left(u_k^{i+\frac{1}{2}} - u_k^{i-\frac{1}{2}}\right) \\ u_{k+1}^{i+\frac{1}{2}} + \frac{1}{2} g \frac{\Delta t}{\Delta x}\left(h_{k+1}^{i+1} - h_{k+1}^i\right) + \frac{1}{2}\Delta t \lambda u_{k+1}^{i+\frac{1}{2}} = u_k^{i+\frac{1}{2}} - \frac{1}{2} g \frac{\Delta t}{\Delta x}\left(h_k^{i+1} - h_k^i\right) - \frac{1}{2}\Delta t \lambda u_k^{i+\frac{1}{2}}. \end{cases}$$

Applying the discretization scheme for the whole estuary with n grid points, taking into account the boundary conditions in Equation 3.13, $h(0,t) = h(t)$ and $u(L,t) = 0$, results into the set of equations:

$$\mathbf{A}\mathbf{x}_{k+1} = \mathbf{B}\mathbf{x}_k + \mathbf{c}_k \tag{3.16}$$

where $\mathbf{c}_k$ denotes the open boundary condition as $\mathbf{c}_k = [h(k), 0, 0, ..., 0]^T$, and the matrices $\mathbf{A}$ and $\mathbf{B}$ of dimensions $2n \times 2n$ are given as:

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & ... & 0 \\ -\mathtt{t1} & 1+\mathtt{t2} & \mathtt{t1} & 0 & 0 & ... & 0 \\ 0 & -\mathtt{t3} & 1 & \mathtt{t3} & 0 & ... & 0 \\ \vdots & 0 & -\mathtt{t1} & 1+\mathtt{t2} & \mathtt{t1} & \ddots & 0 \\ 0 & \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \vdots & \ddots & -\mathtt{t1} & 1+\mathtt{t2} & \mathtt{t1} \\ 0 & 0 & 0 & ... & 0 & 0 & 1 \end{pmatrix}$$

and

$$\mathbf{B} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & ... & 0 \\ \mathtt{t1} & 1-\mathtt{t2} & -\mathtt{t1} & 0 & 0 & ... & 0 \\ 0 & \mathtt{t3} & 1 & -\mathtt{t3} & 0 & ... & 0 \\ \vdots & 0 & \mathtt{t1} & 1-\mathtt{t2} & -\mathtt{t1} & \ddots & 0 \\ 0 & \vdots & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \vdots & \ddots & \mathtt{t1} & 1-\mathtt{t2} & -\mathtt{t1} \\ 0 & 0 & 0 & ... & 0 & 0 & 0 \end{pmatrix}$$

where $\begin{cases} \mathtt{t1} = \frac{1}{2} g \frac{\Delta t}{\Delta x} \\ \mathtt{t2} = \frac{1}{2} \lambda \Delta t \\ \mathtt{t3} = \frac{1}{2} D \frac{\Delta t}{\Delta x} \end{cases}$, and with the right closed boundary condition incorporated in the last row of $\mathbf{B}$.

Solving the set of equations in Equation 3.16 for a new state at a new timestep $k+1$ is accomplished by taking the inverse of $\mathbf{A}$, as

$$\mathbf{x}_{k+1} = \mathbf{A}^{-1}(\mathbf{B}\mathbf{x}_k + \mathbf{c}_k). \tag{3.17}$$

Equation 3.17 is a linear model following the state-space representation introduced in Equation 2.9 in section 2.1, with $\mathbf{M}_k = \mathbf{A}^{-1}\mathbf{B}$ and $\mathbf{B}_k\mathbf{u}_k = \mathbf{A}^{-1}\mathbf{c}_k$. Equation 3.17 allows for sequential computations of waterlevel and velocities by stating an initial state $\mathbf{x}_0$, where all new states $\mathbf{x}_k$ for any time k are immediately followed. Since each new iteration $k+1$ will only depend on the previous time step k, this process is called a Markov process.

3.4. Stochastic model

Hydrodynamic models are stochastic processes, which indicates that noise may be present within the system, already stated in section 2.2. Until now, the deterministic model without noise is derived in section 3.1, section 3.2 and section 3.3. The deterministic model will always have the same estimate when the same initial state is applied. In this section, the ways noise may be present in the system is described.

Forms of noise may include: model noise, initial state noise and measurement noise. Model noise is defined by the inaccuracy of the model, initial state noise by a poorly chosen initial state and measurement noise by measurement errors in observations. These may be only at one instant, spatial correlated or time correlated. Spatial correlated noise is described by a covariance matrix $\mathbf{\Gamma}$ defined by coefficients $\gamma(x_i, x_j)$ as

$$\gamma(x_i, x_j) = ae^{-b|x_i - x_j|}, \tag{3.18}$$

with $a, b \in \mathbb{R}$. Then the noise for each grid cell is determined by the Cholesky decomposition as:

$$\mathbf{w}_k = \mathbf{\Gamma}^{\frac{1}{2}} \mathcal{N}(0, \mathbf{I}), \tag{3.19}$$

where the Cholesky decomposition of a real symmetric positive-definite matrix $\mathbf{A}$ is defined as the decomposition of the matrix $\mathbf{A}$ into

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T, \tag{3.20}$$

with $\mathbf{L}$ a lower triangular matrix where the diagonal entries are real and positive. This kind of noise defined in Equation 3.19 is large when two grid cells are close to eachother.

For time correlation, the noise is time correlated and modelled with an first-order autoregressive process or AR(1) model on the open boundary condition. This will simulate noise on the boundary condition due to not knowing the boundary condition with certainty. To that end,

$$h_k^1 = h(k) + N_k,$$

where

$$N_{k+1} = \alpha N_k + w_k,$$

with $\alpha = \exp(-\Delta t/T)$, with T the time period of the water flowing in the boundary and w_k white noise distributed as $w_k \sim \mathcal{N}(0, \sigma_w^2)$ and $N_0 = 0$. The variance σ_w^2 can be chosen according to what the variance of N_k, σ_N^2 should be. Since

$$\begin{aligned}\sigma_N^2 = \mathbb{E}(N_{k+1}^2) &= \mathbb{E}\big((\alpha N_k + w_k)^2\big) \\ &= \mathbb{E}(\alpha^2 N_k^2 + 2\alpha N_k w_k + w_k^2) \\ &= \alpha^2 \mathbb{E}(N_k^2) + 2\alpha \mathbb{E}(N_k)\mathbb{E}(w_k) + \sigma_w^2 \\ &= \frac{\sigma_w^2}{1-\alpha^2},\end{aligned} \tag{3.21}$$

where is used that $\mathbb{E}(N_k^2) = \sigma_N^2$, w_k has zero mean and N_k and w_k are independent. Therefore, $\sigma_w = \sigma_N\sqrt{1-\alpha^2}$ and thus the variance of N_k depends on the variance of w_k. Upon adding the AR(1) forcing term to our sequential model in Equation 3.17, a problem occurs. Since N_k is depending on all previous N_l with $0 < l < k$, the model will not be a Markov process anymore. To be able to apply sequential data assimilation algorithms on this model in later chapters, the model needs to be a Markov process. To attain this problem, the AR(1) forcing is taken to be a part of the state vector as:

$$\mathbf{x}_k = \left(h_k^1, u_k^{1+\frac{1}{2}}, h_k^2, u_k^{2+\frac{1}{2}}, \dots, u_k^{n-\frac{1}{2}}, h_k^n, u_k^{n+\frac{1}{2}}, N_k\right)^T.$$

Therefore, the model in Equation 3.16 is updated as:

$$\mathbf{A}^*\mathbf{x}_{k+1} = \mathbf{B}^*\mathbf{x}_k + \mathbf{c}_k + \mathbf{w}_k \tag{3.22}$$

where $\mathbf{w}_k = (0, 0, 0, \dots, 0, 0, w_k)^T$ and

$$\mathbf{A}^* = \begin{pmatrix} & & & & & -1 \\ & & & & & 0 \\ & & \mathbf{A} & & & 0 \\ & & & & & \vdots \\ & & & & & 0 \\ 0 & 0 & \dots & 0 & 0 & 1 \end{pmatrix}; \qquad \mathbf{B}^* = \begin{pmatrix} & & & & & 0 \\ & & & & & 0 \\ & & \mathbf{B} & & & 0 \\ & & & & & \vdots \\ & & & & & 0 \\ 0 & 0 & \dots & 0 & 0 & \alpha \end{pmatrix}.$$

To arrive at the sequential Markov process:

$$\mathbf{x}_{k+1} = (\mathbf{A}^*)^{-1}(\mathbf{B}^*\mathbf{x}_k + \mathbf{c}_k + \mathbf{w}_k). \tag{3.23}$$

To summarize, the sequential 1D shallow water model for an estuary with one open boundary condition and one closed boundary condition is given in algorithm 1.

Algorithm 1: Stochastic Hydrodynamic shallow water sequential model for $k = 0$ until $k = T$. For the computation of $\mathbf{A}^*$ and $\mathbf{B}^*$, the settings for g, λ, D, Δt, Δx, n and α are needed.

Result: $\mathbf{x}_k$ for all k ;
Initialize: $\mathbf{x}_0$;
Input: $\mathbf{x}_{k-1}$, $\mathbf{A}^*$, $\mathbf{B}^*$, $\mathbf{w}_{k-1}$, $\mathbf{c}_{k-1}$;
for *k = 1 to T* **do**
  $\mathbf{y}_{k-1} = \mathbf{B}^*\mathbf{x}_{k-1} + \mathbf{c}_{k-1} + \mathbf{w}_{k-1}$;
  $\mathbf{x}_k = (\mathbf{A}^*)^{-1}\mathbf{y}_{k-1}$;
end

3.4.1. Ensemble model

Since stochastic models possess randomness and therefore result in different model estimates for runs of the same model, examining multiple of these estimates and their differences will explain a lot about the behavior of the stochastic model. A set of possible estimates of a stochastic model is called an ensemble, and generating this ensemble is called a Monte-Carlo simulation, see Figure 3.4.

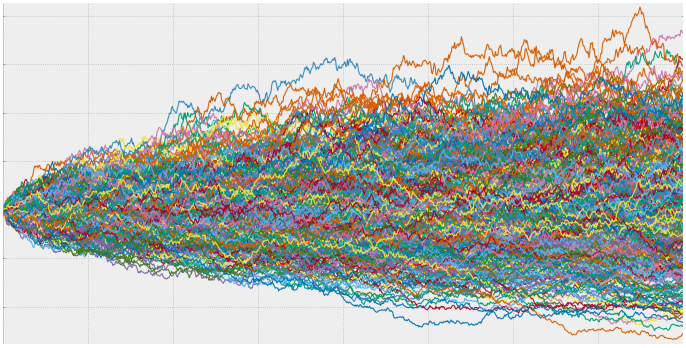


Figure 3.4: A Monte-Carlo simulation. A stochastic model is modelled by taking an ensemble of the model distribution. The mean of the ensemble is used as prediction, while the covariance of the ensemble will represent the variance of the prediction. Figure retrieved from [69].

The hydrodynamic model is a stochastic process. Therefore, by performing multiple model simulations, information is obtained about the spread of the model. To that end, assume an ensemble $\boldsymbol{\xi}_k = \left(\mathbf{x}_k^1, \mathbf{x}_k^2, \ldots, \mathbf{x}_k^N\right)$ of N different stochastic model estimates or ensemble members at time k. The ensemble mean $\hat{\boldsymbol{\xi}}_k$ at a time step k represents the total averaged estimate as:

$$\hat{\boldsymbol{\xi}}_k = \frac{1}{N}\sum_i \mathbf{x}_k^i = \boldsymbol{\xi}_k \mathbf{1}_{N\times N}, \tag{3.24}$$

where $\mathbf{1}_{N\times N}$ is a square matrix of dimension $N \times N$ where each element is equal to $\frac{1}{N}$. The covariance of the ensemble $\mathbf{P}_k = \mathrm{Cov}(\boldsymbol{\xi}_k, \boldsymbol{\xi}_k)$ shows the spread in ensemble. Applying the ensemble theory at the earlier defined stochastic hydrodynamic model in algorithm 1, will give the algorithm in algorithm 2.

Algorithm 2: Stochastic Hydrodynamic shallow water sequential ensemble model for $k = 0$ until $k = T$. For the computation of $\mathbf{A}^*$ and $\mathbf{B}^*$, the settings for g, λ, D, Δt, Δx, n and α are needed.

Result: $\hat{\boldsymbol{\xi}}_k, \mathbf{P}_k$ for all k ;
Initialize: $\boldsymbol{\xi}_0$;
Input: $\boldsymbol{\xi}_{k-1}$, $\mathbf{A}^*$, $\mathbf{B}^*$, $\mathbf{W}_{k-1}$, $\mathbf{C}_{k-1}$;
for *k = 1 to T* **do**
  $\mathbf{Y}_{k-1} = \mathbf{B}^*\boldsymbol{\xi}_{k-1} + \mathbf{C}_{k-1} + \mathbf{W}_{k-1}$;
  $\boldsymbol{\xi}_k = (\mathbf{A}^*)^{-1}\mathbf{Y}_{k-1}$;
  $\hat{\boldsymbol{\xi}}_k = \boldsymbol{\xi}_k \mathbf{1}_{N\times N}$;
  $\mathbf{E}_k = \boldsymbol{\xi}_k - \boldsymbol{\xi}_k \mathbf{1}_{N\times N}$;
  $\mathbf{P}_k = \frac{1}{N-1}\mathbf{E}_k(\mathbf{E}_k)^T$;
end

In this algorithm, $\mathbf{W}_k \in \mathbb{R}^{n\times N}$ denotes the matrix with Gaussian white noise, $\mathbf{C}_k$ is defined as the matrix with boundary conditions for every ensemble member and $\mathbf{P}_k$ equals the error covariance matrix.

An example run of the 1D model explained in section 6.1 with $N = 50$ ensembles at Vlissingen is depicted in Figure 3.5 for waterlevel and 3.6 for water velocity.

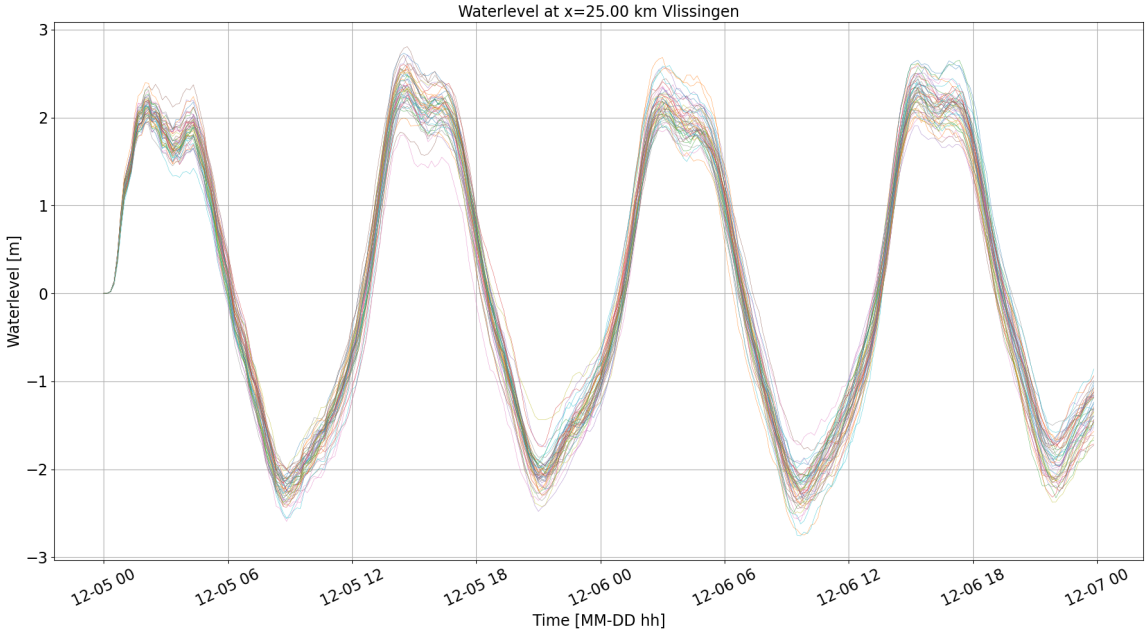


Figure 3.5: An ensemble of 50 waterlevel estimates at Vlissingen, modelled stochastically with algorithm 2 with settings in Table 6.1 over 2 days. The mean of this ensemble results into the prediction of the waterlevel.

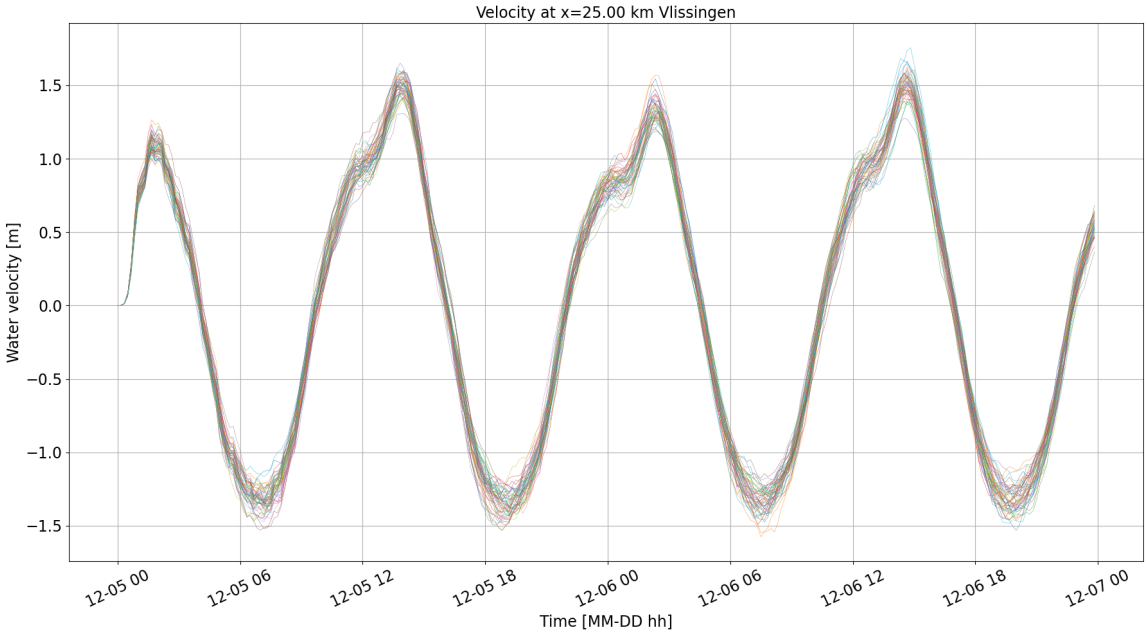


Figure 3.6: An ensemble of 50 water velocity estimates at Vlissingen, modelled stochastically with algorithm 2 with settings in Table 6.1 over 2 days. The mean of this ensemble results into the prediction of the velocity.

3.5. Delft3D FM for 2D and 3D nonlinear models

For the higher-dimensional and nonlinear hydrodynamic models, this thesis will focus on software packages instead of analytical differential equations. The software package considered is Delft3D FM, since it is used in many sortlike applications (see subsection 1.3.2) and it is developed within Deltares. Delft3D FM or Delft3D Flexible Mesh Suite [24] is a modelling environment software package for the modelling of hydrodynamical situations made by Deltares combining earlier versions of other hydrodynamical modelling software, like Delft3D 4 [23] and SOBEK [25]. It is useful for a broad range of

applications, like simulation of storm surges, hurricanes, tsunamis and coastal, river or urban floods. The software package produces detailed waterlevels and flows, waves, sediment transports, water quality and ecology, morphology maps and can simulate the interactions between all of these different processes. These are incorporated into the state vector. The usage of Delft3D FM instead of other software packages is stimulated by high flexibility in the software: the software supports 1D/2D/3D calculations, flexible creation of unstructured grids unlike previous versions, has high computational performance and allows for parallel computations on high-performance computing clusters [55]. D-Flow FM, the software engine of Delft3D FM, runs up to 20 times faster than previous SOBEK FLOW engine of Deltares modelling software SOBEK 2 and is faster than Delft3D-FLOW engine of Delft3D 4 as well [55].

The exact model solved within Delft3D FM is a very extensive set of partial differential equations and is described in chapter 8 through 14 of [28], respectively for hydrodynamics, transport of matter, 3D modelling, heat transport, wind, hydraulic structures and bedforms and vegetation. These equations are solved using discretization techniques, making use of Gauss elimination based on minimum degree, Conjugate Gradient method and the applied finite element method, further explained in [29]. Depending on the situation, some unnecessary models can be left out, such that the resulting model will represent the situation best and will reduce computational time. In this thesis, the model is limited to include hydrodynamics, 3D modelling, heat transport and wind.

The usage of Delft3D FM and techniques behind the models are all explained within the manuals [26, 28–31], while an introduction for the usage of the software is given in an introductory exercise course constructing the model with 2D grid of the Western Scheldt [22]. In Figure 3.7, the GUI of Delft3D FM is depicted with an example of the modelled waterlevel within the Western Scheldt, retrieved from this course.

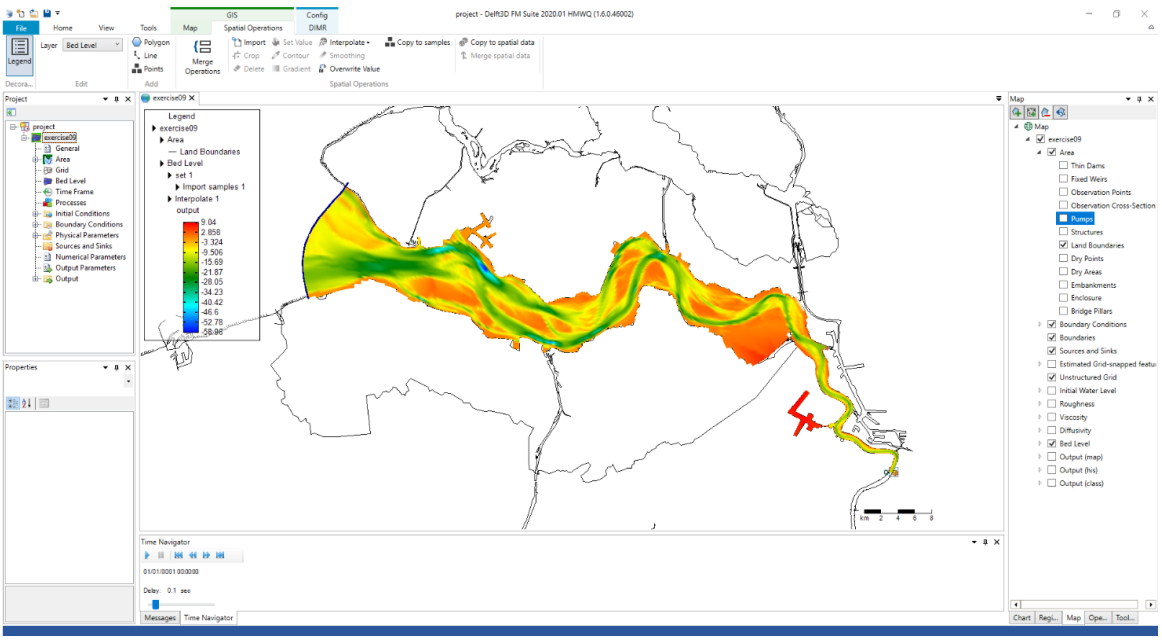


Figure 3.7: Graphical User interface of Delft3D FM with plotted the bedlevel within the Western Scheldt, part of the introductory exercises course on using Delft3D FM from Deltares [22].

Delft3D FM works with a main input file where all model parameters, involved processes and in- and output files for the specific model are described, see Table 3.1 for the most important settings. This file is the Master Definition Unstructured file (MDU) file. Furthermore, extra input as e.g. the grid for the model, boundary conditions, measuring stations and wind can be set by inputting the necessary files. Upon running a model, a nonlinear state-space representation is obtained, where multiple physical variables can be estimated. An example of the waterlevel in the Western Scheldt is given in Figure 3.8.

Table 3.1: For this thesis relevant model in- and output settings and input files of the Delft3D FM MDU file. All other parameters are set to their default values, found in [28].

Setting	Unit	Description
WaterLevIni	m	Initial water level at the whole grid
Bedlevuni	m	Uniform bedlevel for the whole grid
Bedslope	-	Bedslope inclination, as $z =$ bedlevuni $+ x *$ bedslope
AngLat	-	Angle of latitude S-N °, (0 = no Coriolis force)
AngLon	-	Angle of longitutde E-W °, (0 = GMT+0)
Kmx	-	Number of vertical layers
Layertype	-	Type of vertical layers (1 = σ-layers, 2 = z-layers)
Vicouv	$\frac{m^2}{s}$	Uniform horizontal eddy viscosity
Dicouv	$\frac{m^2}{s}$	Uniform horizontal eddy diffusivity
TidalForcing	-	Tidal forcing (0 = no, 1 = yes)
Temperature	-	Temperature model (0 = no, 5 = heat-flux model)
InitialTemperature	°C	Initial temperature level for the whole grid
Secchidepth	m	Water clarity parameter
Stanton	-	Coefficient of convective heat flux
Dalton	-	Coefficient of evaporative heat flux
RefDate	-	Reference date at initial time step as YYYYMMDD
Tunit	-	Time unit in MDU file (D, h, m or s)
DtUser	s	Timestep in for interval ext forcing and his/map-file
TStart	Tunit	Start time of calculation w.r.t. RefDate
TStop	Tunit	Stop time of calculation w.r.t. RefDate
WindExt	-	Include wind (0 = no, 2 = yes)
HisInterval	s	Time interval of model output at the measuring stations
MapInterval	s	Time interval of model output at every grid cell
Wrihis_temperature	-	Write temperature output to his file (0 = no, 1 = yes)
Wrimap_waterlevel_s0	-	Write old waterlevel s0 output to map file (0 = no, 1 = yes)
Wrimap_waterlevel_s1	-	Write new waterlevel s1 output to map file (0 = no, 1 = yes)
Wrimap_velocity_component_u0	-	Write old velocity component to map file (0 = no, 1 = yes)
Wrimap_velocity_component_u1	-	Write new velocity component to map file (0 = no, 1 = yes)
Wrimap_velocity_vector	-	Write velocity vector to map file (0 = no, 1 = yes)
Wrimap_flow_flux_q1	-	Write flux q1 to map file (0 = no, 1 = yes)
Wrimap_heat_fluxes	-	Write heat fluxes to map file (0 = no, 1 = yes)
Wrimap_temperature	-	Write temperature to map file (0 = no, 1 = yes)

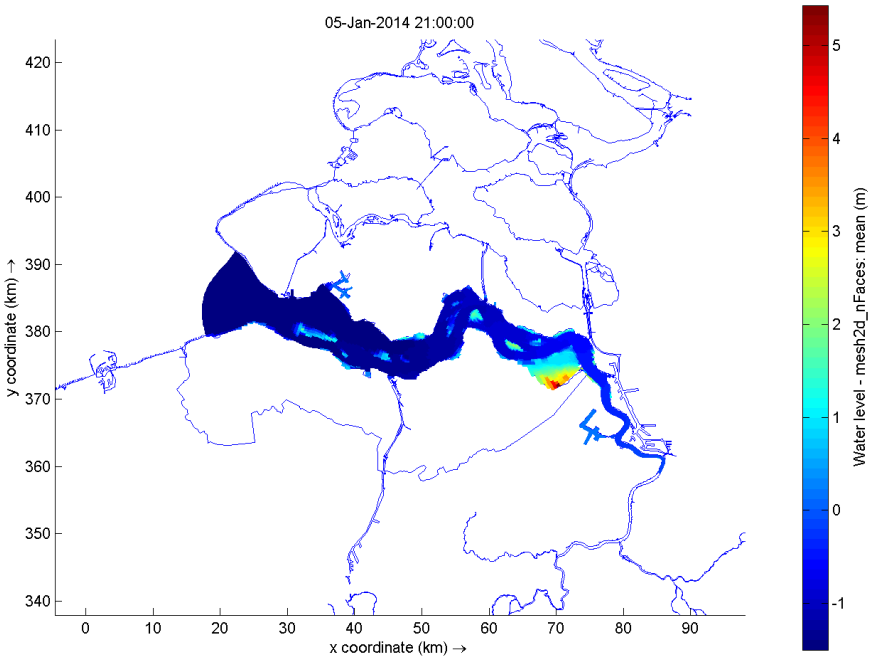


Figure 3.8: The waterlevel estimate in the Western Scheldt at 2014-01-05 21:00, computed with Delft3D FM, part of the introductory course on using Delft3D FM from Deltares [22].

3.5.1. Grid in 2D and 3D

The hydrodynamic model described in Delft3D FM uses both structured and unstructured grids for the xy-plane. Structured grids are grids with implicit connectivity which structure allows for simple identification of nodes and edges, see left figure in Figure 3.9. Unstructured grids are grids with more random assigned nodes and edges, see right figure in Figure 3.9. Both structured and unstructured

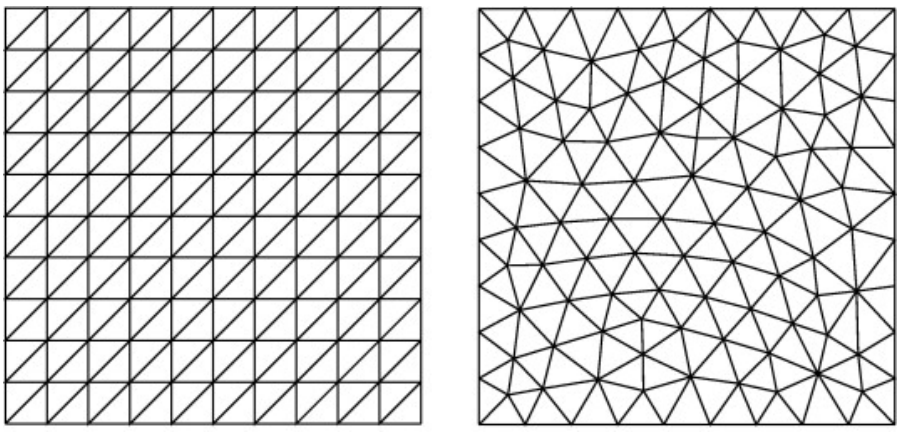


Figure 3.9: Left: structured triangular grid. Right: unstructured triangular grid. Figure retrieved from [43].

grids allow for changes in resolution size. The advantage of unstructured grids compared to structured grids is that it is easier to locally refine the grids to for example follow curved boundaries, and therefore easier to create grids for areas with complex structure that completely cover the area. An example of a 2D computational grid for the Western Scheldt estuary is given in Figure 3.10.

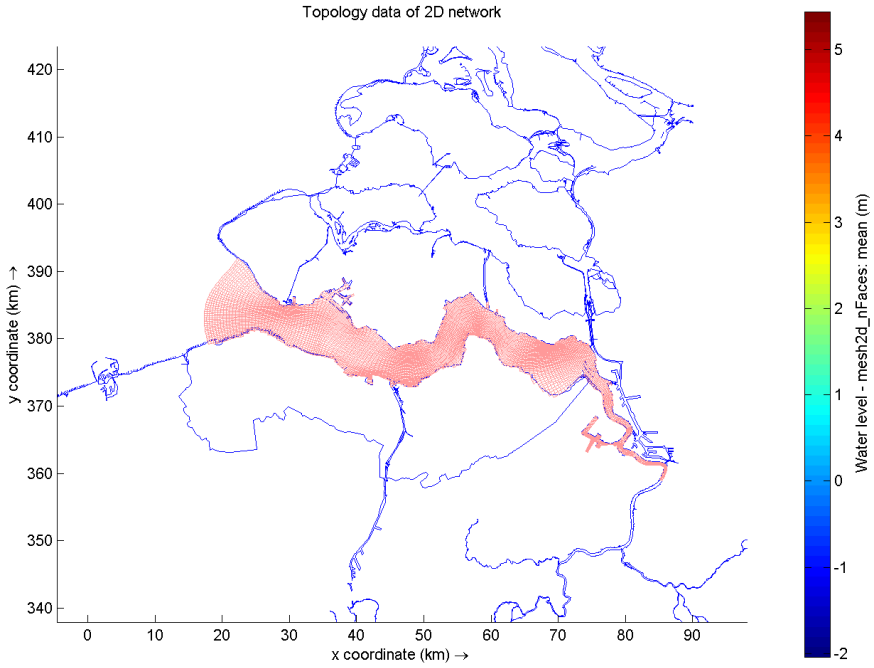


Figure 3.10: The computational grid in the Western Scheldt at 2014-01-05 21:00, created with Delft3D FM, part of the introductory course on using Delft3D FM from Deltares [22].

For vertical layering, the more complex models in 2D and 3D are able to use a cartesian grid and so called σ-layers using the σ-coordinate system. The σ-coordinate system is often used describing the depth profile in oceanics. The grid of the σ-coordinate system consists of layers in between two so-called σ planes with $\sigma = 0$ and $\sigma = 1$, see the left figure in Figure 3.11. These planes follow the bedlevel and the moving free surface of the water exactly. The layers in between are evenly spaced in the z-direction, and therefore a smooth grid is presented, where all depth information is present within the grid and a higher resolution is present in shallower areas. For a grid with 20 σ-layers, the layers are divided as in Table 3.2. In comparison to a normal Cartesian x, y, z coordinate system, see the right figure in Figure 3.11, the grid consists of straight lines and is orthogonal. In this way, a bedlevel which is not straight will not be accurately represented by the grid.

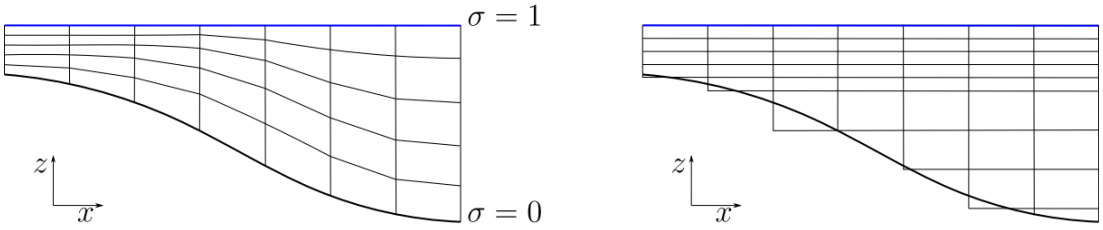


Figure 3.11: Left: Depth profile with σ-coordinate system (σ-model). Right: Depth profile with Cartesian z-coordinate system (Z-model). Figure retrieved from [29].

Table 3.2: σ-value, number of the σ-layer and position within the water column for 20 σ-layers.

σ	σ-layer	Position
1	-	Water surface
0.975	20	
0.925	19	
0.875	18	
0.825	17	
0.775	16	
0.725	15	
0.675	14	
0.625	13	
0.575	12	
0.525	11	
0.475	10	
0.425	9	
0.375	8	
0.325	7	
0.275	6	
0.225	5	
0.175	4	
0.125	3	
0.075	2	
0.025	1	
0	-	Bedlevel

4

Data assimilation

The mathematical framework for data assimilation models is described in section 4.1, 4.2 and 4.3 and software performing data assimilation is described in section 4.4. Most information in this chapter is obtained from [61], [60] and [53] and other sources are mentioned.

4.1. What is data assimilation?

Data assimilation is, as mentioned by [58, p.73]: “the process of finding the model representation which is most consistent with the observations.” The model representation is defined by a state vector, already explained in section 2.1.

The goal of data assimilation may differ. Most often the goal is defined as: the optimal state estimate of a system is sought for, which will represent the true state of a system best. This is also the goal in this thesis. Other goals may include: optimal determination of initial conditions of a model, interpolation of sparse observation data using knowledge of the system to estimate on the whole domain, optimizing numerical model parameters based on observations and estimating unobserved quantities.

There exist two branches of data assimilation methods, as mentioned in section 1.3: variational and sequential methods. Variational methods, like 3D-Var and 4D-Var, seek to find a state that minimizes a cost function J depending on a determined difference (mainly least square error) between the observations and model solution over a given time interval. This allows to solve the global problem immediately. However, these methods often do not converge [60] and are difficult to implement since these methods require that both the linearized and their adjoint models (backward in time) are implemented.

Sequential methods, like the Kalman filter and Ensemble Kalman filter, propagate the model and correct the model state estimate at times when observations are available. The procedure of sequential data assimilation methods is shown in Figure 4.1. The process starts with the known model, in this thesis the hydrodynamic model. The model progresses one time step, also called the forecast or background step, where no observations are included yet. Subsequent to the forecast step, the data assimilation method is applied. The results of the forecast step are updated with the observations via a metric resulting in the state update or analysis step, or also known as the assimilation step, which lie closer to the true state of the system, see Figure 4.1 right. The result then is again input for the model to progress one time step further until the stopping time of the model is reached.

This thesis focuses on the use of sequential data assimilation methods for forecasting systems, since sequential methods not only provide state estimates but also explicit error estimates and thus give insight into the accuracy of the method. Furthermore, sequential methods are used often for similar applications sought for in this thesis, see subsection 1.3.2. Moreover, the use of variational methods is discouraged due to their complex implementation.

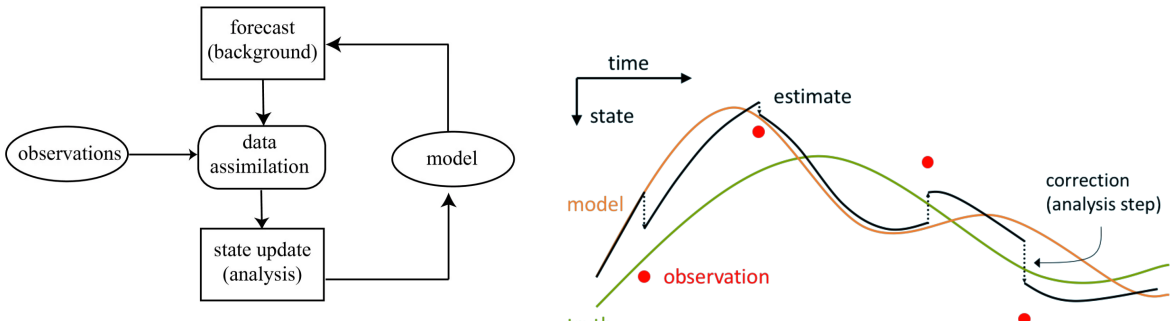


Figure 4.1: Procedure of sequential data assimilation. Left: Schematic. Figure retrieved from [50]. Right: Updated state estimation in time with model state in orange, true state in green, observations in red and assimilated state in black. Figure retrieved from [18].

4.2. Kalman Filter for linear systems

The Kalman filter (KF) is a linear minimum-error-variance sequential data assimilation algorithm. The algorithm optimally combines model predictions and observations to achieve more accurate forecast values and will determine error estimates of the result. The Kalman Filter operates as all sequential data assimilation algorithms, see section 4.1, in two steps: the forecast step, where the next model time iteration is performed and assimilation step, where the result of the forecast step is updated with observations.

The Kalman filter adapts the state-representation notation from section 2.1. To that end, consider a model state vector $\mathbf{x}_k$ with observations $\boldsymbol{z}_k$ at time k, defined as

$$\begin{cases} \mathbf{x}_k = (x_k^1, \dots, x_k^n)^T \\ \boldsymbol{z}_k = (z_k^1, \dots, z_k^m)^T, \end{cases} \tag{4.1}$$

with $x_k^1 \dots x_k^n$ state variables and $z_k^1 \dots z_k^m$ observations of specific state variables. Note that the dimension of the state vector n is generally much larger than the dimension of the observation vector m, due to the sparse amount of observations available. Consider both deterministic and stochastic models, see section 2.2. For the deterministic case, the state vector is progressed in time k and related to the observations with output matrix $\mathbf{H}_k$ as:

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{M}_k \mathbf{x}_k + \mathbf{B}_k \mathbf{u}_k \\ \boldsymbol{z}_k = \mathbf{H}_k \mathbf{x}_k, \end{cases} \tag{4.2}$$

where the model matrix $\mathbf{M}_k$ and the observation operator matrix $\mathbf{H}_k$, may be time-dependent. The stochastic model representation is described as:

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{M}_k \mathbf{x}_k + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k \\ \boldsymbol{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k, \end{cases} \tag{4.3}$$

where $\mathbf{w}_k$, $\mathbf{v}_k$ are Gaussian white noise, of respectively positive definite, symmetric and possibly time-dependent covariance matrices $\mathbf{Q}_k = \mathrm{Cov}(\mathbf{w}_k)$ and $\mathbf{R}_k = \mathrm{Cov}(\mathbf{v}_k)$, thus $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k)$ and $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$. Within the Kalman filter, the optimal trade-off between model prediction $\mathbf{x}_k$ and observations $\boldsymbol{z}_k$ is made at every time step. The algorithm is presented in algorithm 3.

Algorithm 3: Kalman Filter. The Kalman Filter assumes a stochastic model and solves for the optimal $\mathbf{x}_k^a$ given the observations $\boldsymbol{z}_k$ at each timestep k in the forecast and analysis or assimilation step.

Result: $\mathbf{x}_k^a$, $\mathbf{P}_k^a$;
Input: $\mathbf{x}_0^a, \mathbf{P}_0^a, \mathbf{u}_k, \boldsymbol{z}_k$;
for *k = 1 to T* **do**
  Forecast step:
  $\mathbf{x}_k^f = \mathbf{M}\mathbf{x}_{k-1}^a + \mathbf{B}\mathbf{u}_k$;
  $\mathbf{P}_k^f = \mathbf{M}\mathbf{P}_{k-1}^a\mathbf{M}^T + \mathbf{Q}_k$;
  Assimilation step:
  $\mathbf{x}_k^a = \mathbf{x}_k^f + \mathbf{K}_k(\boldsymbol{z}_k - \mathbf{H}_k\mathbf{x}_k^f)$;
  $\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^f$;
end

In the forecast step, the model prediction $\mathbf{x}_k^f$ and model error covariance $\mathbf{P}_k^f$ gets forwarded to the next timestep k. The assimilation step calculates the optimal trade-off for the assimilated or analysis prediction $\mathbf{x}_k^a$ and analysis error covariance $\mathbf{P}_k^a$. It is done via the Kalman gain $\mathbf{K}_k$, which determines by how much the previous forecast should be updated, based on the error between the forecast $\mathbf{x}_k^f$ and the observations $\boldsymbol{z}_k$. The analysis error covariance $\mathbf{P}_k^a$ is defined via Equation A.2 in Appendix A.
The filter is optimal when the model error covariance $\mathbf{P}_k^a$ is minimized. This happens when the trace of $\mathbf{P}_k^a$ is minimized. Since by Equation A.1 in Appendix A:

$$\mathbf{P}_k^a = \mathbf{P}_k^f - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^f - \mathbf{P}_k^f\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)\mathbf{K}_k^T, \tag{4.4}$$

minimization of $\mathbf{P}_k^a$ results into

$$\frac{\partial \text{tr}(\mathbf{P}_k^a)}{\partial \mathbf{K}_k} = 0 \implies -2(\mathbf{H}_k\mathbf{P}_k^f)^T + 2\mathbf{K}_k(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k) = 0,$$

and thus an optimal Kalman gain $\mathbf{K}_k$ for

$$\mathbf{K}_k = \mathbf{P}_k^f\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)^{-1}. \tag{4.5}$$

Intuitively, to notice the trade-off between the model error covariance $\mathbf{P}_k^f$ and the observation error covariance $\mathbf{R}_k$, in the scalair case with one observation, the Kalman Gain can be written as ($H_k = 1$ in this case):

$$K_k = \frac{P_k^f}{P_k^f + R_k}.$$

When the observation error covariance R_k becomes very large in comparison to P_k^f, then by intuition:

$$\lim_{R_k \to \infty} K_k = \lim_{R_k \to \infty} \frac{P_k^f}{P_k^f + R_k} = 0$$

and then the assimilation step in the Kalman filter in algorithm 3 reduces to:

$$\begin{cases} x_k^a = x_k^f \\ P_k^a = P_k^f. \end{cases}$$

Thus, the filter will completely ignore the presence of observations and will only consider the model results in the optimization process. Similarly, when P_k^f becomes very large in comparison to R_k, then

$$\lim_{P_k^f \to \infty} K_k = \lim_{P_k^f \to \infty} \frac{P_k^f}{P_k^f + R_k} = \lim_{P_k^f \to \infty} \frac{1}{1} = 1,$$

where l'Hôpital's rule is used for the calculation of the limit. Consequently the assimilation step in the Kalman filter in algorithm 3 reduces by using Equation 4.4 to

$$\begin{cases} x_k^a = z_k, \\ P_k^a = R^k, \end{cases}$$

resulting in a filter that will completely ignore the model results and will only update the estimate with observations. Therefore, the ratio between the model error covariance and the observation error covariance should not be too large in order to take into account for both model and observations.
Upon applying the Kalman filter, the algorithm stores the full model error covariances. When the dimensions of these matrices are very large, as is the case with hydrodynamic models with many state variables, the memory required to perform the Kalman filter is very high.

4.3. Ensemble Kalman Filter for nonlinear systems

The Ensemble Kalman filter (EnKF) is a sequential data assimilation algorithm, based on Monte-Carlo methods and the Kalman filter mentioned in section 4.2. Monte-Carlo methods are methods used to model the probability of different outcomes in a stochastic model, already mentioned in subsection 3.4.1. Monte-Carlo methods examine an ensemble of possible estimates of a stochastic model with known distributions, where the average results in the estimate.
The Ensemble Kalman filter is a Monte-Carlo approximation of the Kalman filter, in a way that the model error covariances are approximated by a sample error covariances. To that end, the model error covariances are not fully computed, resulting in a decreased computational time and memory needed to perform this algorithm in comparison to the Kalman filter. The Ensemble Kalman filter is an extension to the Kalman filter in the way that it can be used for nonlinear systems. Again, assume a model state vector $\mathbf{x}_k$ with observations $\mathbf{z}_k$ as defined in section 2.1:

$$\begin{cases} \mathbf{x}_k = (x_k^1, \dots, x_k^n)^T \\ \mathbf{z}_k = (z_k^1, \dots, z_k^m)^T. \end{cases} \tag{4.6}$$

Assume an ensemble of N of these model state vectors $\boldsymbol{\xi}_k \in \mathbb{R}^{n \times N}$ and assume an ensemble of observations $\mathbf{Z}_k \in \mathbb{R}^{m \times N}$ defined as:

$$\begin{cases} \boldsymbol{\xi}_k = (\mathbf{x}_k^1, \dots, \mathbf{x}_k^N) \\ \mathbf{Z}_k = (\mathbf{z}_k^1, \dots, \mathbf{z}_k^N), \end{cases} \tag{4.7}$$

where

$$\mathbf{z}_k^i = \mathbf{z}_k + \mathbf{e}^i, \tag{4.8}$$

with $i = 1, \dots, N$ and $\mathbf{e}_k^i$ a perturbation on the observations. The perturbations are assembled in a matrix $\boldsymbol{\Upsilon}_k \in \mathbb{R}^{m \times N}$:

$$\boldsymbol{\Upsilon}_k = (\mathbf{e}_k^1, \dots, \mathbf{e}_k^N). \tag{4.9}$$

The ensemble perturbation matrix is given as

$$\mathbf{E}_k = \boldsymbol{\xi}_k - \hat{\boldsymbol{\xi}}_k = \boldsymbol{\xi}_k(\mathbf{I} - \mathbf{1}_{N \times N}). \tag{4.10}$$

The model and observation error covariance matrices from the Kalman filter in algorithm 3 are estimated via the perturbations on the ensembles as

$$\mathbf{P}_k^f \approx \mathbf{P}_k^e = \frac{1}{N-1}\mathbf{E}_k(\mathbf{E}_k)^T \tag{4.11}$$

and

$$\mathbf{R}_k \approx \mathbf{R}_k^e = \frac{1}{N-1}\boldsymbol{\Upsilon}_k(\boldsymbol{\Upsilon}_k)^T. \tag{4.12}$$

The nonlinear state-space representation mentioned in section 2.1 and 2.2, for deterministic and stochastic models, is again given as:

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{M}_{k+1}(\mathbf{x}_k, \mathbf{u}_k, k) \\ \mathbf{z}_k = \mathbf{H}_k(\mathbf{x}_k, \mathbf{u}_k), \end{cases} \tag{4.13}$$

and

$$\begin{cases}\mathbf{x}_{k+1} = \mathbf{M}_{k+1}(\mathbf{x}_k, \mathbf{u}_k, k) + \mathbf{w}_k \\ \mathbf{z}_k = \mathbf{H}_k(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{v}_k. \end{cases} \tag{4.14}$$

For an ensemble of estimates, it is given for deterministic and stochastic models as

$$\begin{cases}\boldsymbol{\xi}_{k+1} = \mathbf{M}_{k+1}(\boldsymbol{\xi}_k, \mathbf{U}_k, k) \\ \mathbf{Z}_k = \mathbf{H}_k(\boldsymbol{\xi}_k, \mathbf{U}_k), \end{cases} \tag{4.15}$$

and

$$\begin{cases}\boldsymbol{\xi}_{k+1} = \mathbf{M}_{k+1}(\boldsymbol{\xi}_k, \mathbf{U}_k, k) + \mathbf{W}_k \\ \mathbf{Z}_k = \mathbf{H}_k(\boldsymbol{\xi}_k, \mathbf{U}_k) + \mathbf{V}_k, \end{cases} \tag{4.16}$$

with $\mathbf{U}_k$ input with boundary conditions for every ensemble, $\mathbf{W}_k \in \mathbb{R}^{n\times N}$ and $\mathbf{V}_k \in \mathbb{R}^{m\times N}$ matrices with Gaussian white noise. The analysis update in the Kalman filter is exchanged by an ensemble analysis update as

$$\boldsymbol{\xi}_k^a = \boldsymbol{\xi}_k^f + \mathbf{P}_k^e \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^e \mathbf{H}_k^T + \mathbf{R}_k^e)^{-1} (\mathbf{Z}_k - \mathbf{H}_k \boldsymbol{\xi}_k^f). \tag{4.17}$$

The algorithm for the Ensemble Kalman filter is presented in algorithm 4.

Algorithm 4: Ensemble Kalman Filter. The Ensemble Kalman Filter assumes a stochastic model and solves for the optimal ensemble $\boldsymbol{\xi}_k^a$ given the perturbed observations $\mathbf{Z}_k$ at each time step k in the forecast and analysis or assimilation step.

Result: $\hat{\boldsymbol{\xi}}_k$, $(\mathbf{P}_k^e)^a$;
Initialize: $\boldsymbol{\xi}_0^a$;
Input: $\boldsymbol{\xi}_{k-1}^a$, $\mathbf{M}$, $\mathbf{W}_k$, $\mathbf{U}_k$, $\mathbf{Z}_k$, $\mathbf{R}_k^e$;
for *k = 1 to T* **do**
  Forecast step:
  $\boldsymbol{\xi}_k^f = \mathbf{M}(\boldsymbol{\xi}_{k-1}^a, \mathbf{U}_{k-1}, k) + \mathbf{W}_{k-1}$;
  $\mathbf{E}_k^f = \boldsymbol{\xi}_k^f - \boldsymbol{\xi}_k^f \mathbf{1}_{N\times N}$;
  $\mathbf{P}_k^e = \frac{1}{N-1} \mathbf{E}_k^f (\mathbf{E}_k^f)^T$;
  Assimilation step:
  $\mathbf{K}_k = \mathbf{P}_k^e \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^e \mathbf{H}_k^T + \mathbf{R}_k^e)^{-1}$;
  $\boldsymbol{\xi}_k^a = \boldsymbol{\xi}_k^f + \mathbf{K}_k (\mathbf{Z}_k - \mathbf{H}_k \boldsymbol{\xi}_k^f)$;
  $\hat{\boldsymbol{\xi}}_k = \boldsymbol{\xi}_k^a \mathbf{1}_{N\times N}$;
  $\mathbf{E}_k^a = \boldsymbol{\xi}_k^a - \boldsymbol{\xi}_k^a \mathbf{1}_{N\times N}$;
  $(\mathbf{P}_k^e)^a = \frac{1}{N-1} \mathbf{E}_k^a (\mathbf{E}_k^a)^T$;
end

The algorithm is indeed similar to the Kalman filter in algorithm 3 in section 4.2, as both have a forecast and assimilation step and the same optimal Kalman gain $\mathbf{K}_k$ as trade-off between the model estimates and the observations z_k. However, the model estimates $\mathbf{x}_k$ in algorithm 3 have been replaced by an ensemble $\boldsymbol{\xi}_k = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N)_k$ of N model estimates. Furthermore, the model error covariances of the forecast and assimilation step are approximated and thus replaced by sample error covariances, based on the error between each ensemble member and the mean of the ensemble. $\mathbf{1}_{N\times N}$ is a square matrix of dimension $N \times N$ where each element is equal to $\frac{1}{N}$.

While algorithm 4 is more efficient than algorithm 3, when the amount of state variables becomes large, the algorithm has a problem. It calculates the inverse of the sample error covariance $\mathbf{P}_k^e$, which is a very large matrix. The algorithm can be rewritten to a more efficient algorithm, where taking the inverse of a large matrix is abandoned, see algorithm 5. Instead, the inverse of the matrix $\boldsymbol{\Psi}_k$, based on the observation matrix $\mathbf{H}_k$ and Cholesky decomposition matrix $\mathbf{L}_k$ of the sample error covariance, is taken, see Equation 3.20. $\boldsymbol{\Psi}_k$ has reduced dimensions, since it is assumed that the dimension of the

observation space is lower than the dimension of the model space.

Algorithm 5: Ensemble Kalman Filter: Efficient formulation. The Ensemble Kalman Filter assumes a stochastic model and solves for the optimal ensemble $\boldsymbol{\xi}_k^a$ given the perturbed observations $\mathbf{Z}_k$ at each time step k in the forecast and analysis or assimilation step.

Result: $\hat{\boldsymbol{\xi}}_k$;
Initialize: $\boldsymbol{\xi}_0^a$;
Input: $\boldsymbol{\xi}_{k-1}^a$, $\mathbf{M}$, $\mathbf{W}_k$, $\mathbf{U}_k$, $\mathbf{Z}_k$, $\mathbf{R}_k^e$;
for *k = 1 to T* **do**
  Forecast step:
  $\boldsymbol{\xi}_k^f = \mathbf{M}(\boldsymbol{\xi}_{k-1}^a, \mathbf{U}_k, k) + \mathbf{W}_k$;
  $\mathbf{L}_k = \frac{1}{\sqrt{N-1}}\left(\boldsymbol{\xi}_k^f - \boldsymbol{\xi}_k^f \mathbf{1}_{N\times N}\right)$;
  Assimilation step:
  $\boldsymbol{\Psi}_k = \mathbf{H}_k \mathbf{L}_k$;
  $\mathbf{K}_k = \mathbf{L}_k \boldsymbol{\Psi}_k^T (\boldsymbol{\Psi}_k \boldsymbol{\Psi}_k^T + \mathbf{R}_k^e)^{-1}$;
  $\boldsymbol{\xi}_k^a = \boldsymbol{\xi}_k^f + \mathbf{K}_k\left(\mathbf{Z}_k - \mathbf{H}_k \boldsymbol{\xi}_k^f\right)$;
  $\hat{\boldsymbol{\xi}}_k = \boldsymbol{\xi}_k^a \mathbf{1}_{N\times N}$;
end

4.4. OpenDA

OpenDA [5] is an opensource software toolbox written in both JAVA and C for quick generic implementation of data assimilation algorithms. The software is designed to work with building blocks or configuration files, where each can be adapted to fit the purpose of the problem. It was brought to life through the merge of two other data assimilation software platforms: COSTA [78] and DATools [82].
Other possible software packages are PDAF [3], Sequoia [4] and DART [1]. However, the use of OpenDA is opted for since it is developed within Deltares and knowledge about the usage is close by. Furthermore, its modular framework allows for clear implementation and quick adaptations.
The building blocks of this software package are divided into the following components, a method, model and observations, shown in Figure 4.2.

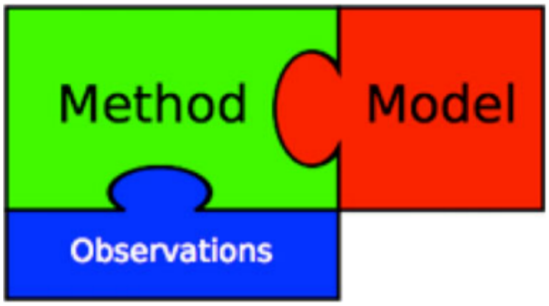


Figure 4.2: OpenDA building blocks, which consist of an overall method, which is using a model and observations. Figure retrieved from [79].

The method is the data assimilation method, where the options are between calibration and data assimilation. Possible calibration algorithms are e.g. DUD and Conjugate Gradient (not discussed further in this thesis) and data assimilation algorithms are e.g. EnKF and 3DVar. These methods are applied to a model, created either in OpenDA or are delivered by external software packages compatible with OpenDA, like Delft3D FM, see section 3.5. To that end, the model in OpenDA or by the external software delivers a deterministic model run to OpenDA. OpenDA is able to combine this deterministic model run and observations in the applied method, to arrive at the sought for application.
How this works into practice is via XML files. These are divided into four categories:

- The main configuration file: an .oda file, specifying the path locations of the other building blocks.

- Stochastic Model: an .xml file specifying the file locations of the model.
- Algorithm: an .xml file specifying which algorithm is used and the settings of this algorithm.
- Stochastic Observer: an .xml file specifying which datasets are used as observations in the algorithm.

When each is specified, the software will automatically derive an optimal state vector prediction with error estimates.
Advantages of OpenDA are the clear implementation with the modular framework with clear distinctions between model, noise, algorithm, and observations, the good performance of OpenDA and the ability to work with small- and large-scale models. Furthermore, since the combination of Delft3D FM and OpenDA is frequently used, the connection between the two software methods already exists and does not need to be implemented. Disadvantages are the high computational time needed for simulations with large ensemble sizes.

5

Experimental setup

This thesis will focus on a start of an operational high-resolutional hydrodynamic and water quality forecasting system for the area of the Belgian pilot of the UNITED project within the North Sea. In particular, this thesis puts the focus on the forecasting of the physical variables waterlevel, water velocity and temperature. The most generally applicable Ensemble Kalman filter data assimilation algorithm is implemented on different hydrodynamic models to investigate the accuracy and computational efficiency of the resulting forecasting system by uncertainty and sensitivity analyses. The goal is to investigate if the application of the filter on the models will result in improved estimates of the models. For this purpose, hyperparameters of the Ensemble Kalman filter are tweaked in experiments to see the difference in performance.

The experimental setup will be discussed in this chapter. In section 5.1, the applicattion of the Ensemble Kalman filter on the investigated models is discussed. Furthermore, in section 5.2, twin experiments to investigate the accuracy and efficiency are discussed and the creation of synthetic data needed for these experiments is discussed. This chapter concludes with an explanation on how to evaluate the estimates in section 5.3.

5.1. Ensemble Kalman filter applied on the hydrodynamic model

In this thesis, multiple variants of the Ensemble Kalman filter on hydrodynamic models in 1D and 2D are investigated to see if the Ensemble Kalman filter improves prediction. Therefore, the application of the Ensemble Kalman filter on the hydrodynamic model for the 1D case is discussed.

The Ensemble Kalman filter has been discussed previously in algorithm 5 in section 4.3. The hydrodynamic model has been discussed in algorithm 2 in section 3.4. The application of the Ensemble Kalman filter on the hydrodynamic model and the resulting algorithm is shown in algorithm 6.

To estimate the performance of algorithm 6 in comparison to the 1D hydrodynamic model without Ensemble Kalman filter described in algorithm 2 in section 3.4, estimates $\hat{\boldsymbol{\xi}}$ produced by both algorithms can be compared. Both algorithms produce error covariances $\mathbf{P}$, thus the spread in estimates can be investigated.

The settings needed for both of the algorithms are presented in Table 5.1. Varying some of these hyperparameters will result in better or worse performing filters, measured by the root mean square error explained in section 5.3. Since the testing of the Ensemble Kalman filter is required, the model settings will not be changed in this thesis. Only the noise settings to change the performance of the Ensemble Kalman filter are varied, in experiments performed in chapter 6.

For the 2D and 3D cases, software is used for the computations, see section 3.5 and section 4.4. Delft3D FM will produce a deterministic estimate, while OpenDA operates on the Delft3D FM software to produce stochastic estimates with or without Ensemble Kalman filter applied. Both estimates can be compared, while noise settings within OpenDA are varied. Experiments are conducted within chapter 7.

Algorithm 6: Hydrodynamic model with applied Ensemble Kalman filter. This algorithm combines algorithm 2 from subsection 3.4.1 and algorithm 5 from section 4.3.

Result: $\hat{\boldsymbol{\xi}}_k$, $\mathbf{P}_k^a$ for all k;
Initialize: $\boldsymbol{\xi}_0^a$;
Input: $\boldsymbol{\xi}_{k-1}^a$, $\mathbf{A}^*$, $\mathbf{B}^*$, $\mathbf{W}_{k-1}$, $\mathbf{C}_{k-1}$, $\mathbf{z}_k$, $\mathbf{R}_k$;
for *k = 1 to T* **do**
  Forecast step:
  $\mathbf{Y}_{k-1} = \mathbf{B}^* \boldsymbol{\xi}_{k-1}^a + \mathbf{C}_{k-1} + \mathbf{W}_{k-1}$;
  $\boldsymbol{\xi}_k^f = (\mathbf{A}^*)^{-1} \mathbf{Y}_{k-1}$;
  $\mathbf{L}_k = \frac{1}{\sqrt{N-1}} \left(\boldsymbol{\xi}_k^f - \boldsymbol{\xi}_k^f \mathbf{1}_{N \times N} \right)$;
  Assimilation step:
  $\boldsymbol{\Psi}_k = \mathbf{H}_k \mathbf{L}_k$;
  $\mathbf{K}_k = \mathbf{L}_k \boldsymbol{\Psi}_k^T (\boldsymbol{\Psi}_k \boldsymbol{\Psi}_k^T + \mathbf{R}_k)^{-1}$;
  $\boldsymbol{\xi}_k^a = \boldsymbol{\xi}_k^f + \mathbf{K}_k \left(\mathbf{Z}_k - \mathbf{H}_k \boldsymbol{\xi}_k^f \right)$;
  $\hat{\boldsymbol{\xi}}_k = \boldsymbol{\xi}_k^a \mathbf{1}_{N \times N}$;
  $\mathbf{E}_k^a = \boldsymbol{\xi}_k^a - \boldsymbol{\xi}_k^a \mathbf{1}_{N \times N}$;
  $\mathbf{P}_k^a = \frac{1}{N-1} \mathbf{E}_k^a (\mathbf{E}_k^a)^T$;
end

Table 5.1: Settings used for the hydrodynamic model with EnKF in algorithm 6, divided in model settings and noise settings.

Parameter	Description
D	Depth
λ	Friction coefficient
g	Acceleration of gravity
n	Number of grid cells
L	Length of the estuary
Δt	Time resolution
T	Total time of the model
t_0	Starting time of the model
Δx	Space resolution
N	Number of ensembles
α	Boundary noise coefficient
σ_N	Model error variance
$\mathbf{R}_k$	Observational error covariance
$\mathbf{W}_k$	Model time-dependent noise
$\boldsymbol{\xi}_0$	Initial ensemble

5.2. Twin experiments

Upon determining the performance of the data assimilation filter in comparison to the model, it is required to know the true state the system is in, as earlier discussed in section 2.3. When the true state vector is known, the estimates of the model with filter and only model are both compared to this true state vector, as will be discussed in section 5.3. Then the performance of both methods is determined. However, the true state the system is in, generally is not known in advance, since both model and model with Ensemble Kalman filter seek to estimate this true state. Sometimes observations are seen as the true state of the system, but these usually do not give a whole true state vector for the whole domain. Therefore, it is not possible to compare and validate estimates on unobserved locations. In twin experiments therefore, the true state is chosen from the stochastically generated model runs, where a fully state vector is computed. Since it is often investigated if data assimilation algorithms will improve model values on unobserved locations, and via a full known true state vector that is possible, twin experiments are useful for validation in data assimilation.

The workflow of a twin experiment is given in Figure 5.1. First, a true state $\mathbf{x}_k^{tr}$ is generated by a stochastic simulation of the model (in red). The estimate of this simulation will assumed to be the true state. Thereafter, from this true state, observations are generated at available measuring locations by adding zero mean white noise to the true state (in blue). The observations to this true state are generated via

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k^{tr} + \mathbf{v}_k, \tag{5.1}$$

where $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$. In this thesis, $\mathbf{R}_k$ is a constant matrix. Notice that the covariance to generate the synthetic observation $\mathbf{R}_k$ is the same as the observation error covariance used in the Ensemble Kalman filter. The observation operator $\mathbf{H}_k$ makes sure that only observations are generated in a measuring station. To that end, the elements of $\mathbf{H}_k$ are constructed in the following way:

$$(\mathbf{H}_k)_{ij} = \begin{cases} 1, \text{if measurement station } i \text{ measures state vector element } \mathbf{x}_k(j) \\ 0, \text{otherwise.} \end{cases} \tag{5.2}$$

It is by choice of the user how many measurement stations and at which locations the measurements are taken, based on the problem at hand. The measurements are assimilated into a simulation of the model to arrive at the assimilated states (also in blue). Lastly, a deterministic simulation of the model is conducted to arrive to the simulated states the model would get without data assimilation (in green). The simulated states or model states and assimilated states are compared with the generated true state by the performance evaluation mentioned in section 5.3.

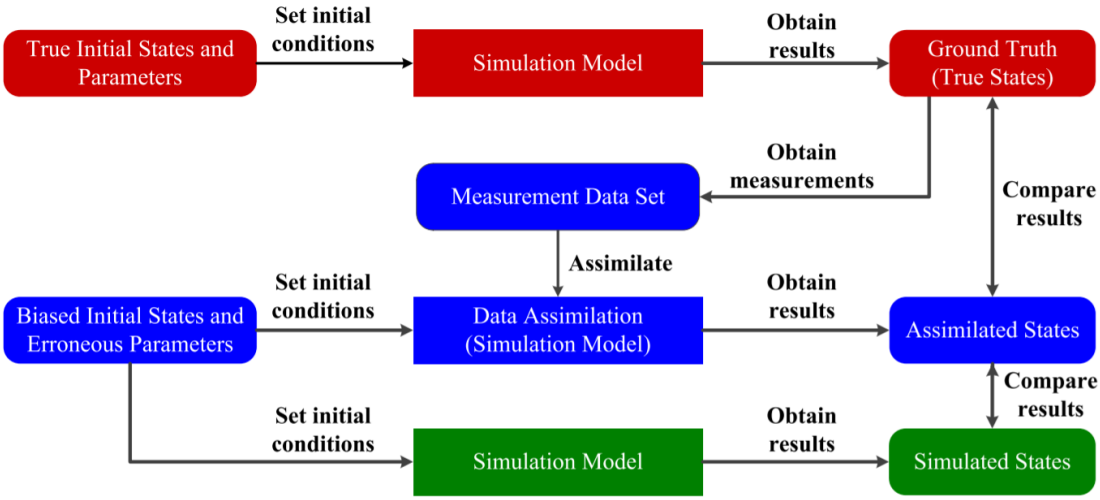


Figure 5.1: The workflow of twin experiments within data assimilation. The red part assumes a stochastic simulation to obtain a generated true state representation of the model. From the true state, a set of measurements is generated. These measurements are assimilated within the blue part of the workflow, to arrive at assimilated states. The green part shows a deterministic simulation of the model, to arrive at the deterministic model estimate. Both the assimilated state estimate and model state estimate are compared to the true state, explained in section 5.3. Figure retrieved from [81].

5.3. Performance evaluation

To validate the estimates from the different algorithms, the root mean square error and the standard deviation of these estimates will be computed. The root mean square error shows the differences between the true state and the ensemble estimates, while the standard deviation shows the spread in the ensemble. Therefore, the deviation from the true state and spread in estimate for each estimate can be compared. Both root mean square error and standard deviation are computed per measurement location within the model. To that end, the root mean square error (rmse) of the model estimate at a time step k is given by

$$\text{rmse}_k = \sqrt{\frac{1}{m}\left(\mathbf{H}_k \mathbf{x}_k^{tr} - \mathbf{H}_k \hat{\boldsymbol{\xi}}_k\right)^T \left(\mathbf{H}_k \mathbf{x}_k^{tr} - \mathbf{H}_k \hat{\boldsymbol{\xi}}_k\right)}. \tag{5.3}$$

with m the amount of measurement locations. To get to the rmse of the algorithm, all rmse_k at each time step is averaged over time. Likewise, the rmse of the estimate at a single measurement location i can be investigated, which is computed as

$$\text{rmse}_{k,i} = \sqrt{\left((\mathbf{H}_k)_i \mathbf{x}_k^{tr} - (\mathbf{H}_k)_i \hat{\boldsymbol{\xi}}_k\right)^T \left((\mathbf{H}_k)_i \mathbf{x}_k^{tr} - (\mathbf{H}_k)_i \hat{\boldsymbol{\xi}}_k\right)}, \tag{5.4}$$

with $(\mathbf{H}_k)_i$ the i-th row of $\mathbf{H}_k$. Again, averaging over time gives the rmse for the estimate at that specific measurement location. The standard deviation of the estimate at a time step k is given as

$$\text{std}_k = \sqrt{\frac{1}{m}\text{tr}\left(\mathbf{H}_k \mathbf{P}_k^f \mathbf{H}_k^T\right)}, \tag{5.5}$$

where the trace of the covariance matrix is taken. To arrive at the std of the whole estimate in time, all std_k at each timestep is averaged over time. The standard deviation of the estimate at a single measurement location i is given as

$$\text{std}_{k,i} = \sqrt{(\mathbf{H}_k)_i \mathbf{P}_k \left((\mathbf{H}_k)_i\right)^T}. \tag{5.6}$$

Averaging over time gives the standard deviation for the measurement location i. Both Equation 5.3 and Equation 5.5 are taken from [61].

6

1D model

In this chapter, the twin experiments mentioned in chapter 5 are performed with the Ensemble Kalman filter (section 4.3) on the 1D hydrodynamic model (section 3.1) for a simplified Western Scheldt estuary model. The results in this chapter are to better understand the workings of the Ensemble Kalman filter applied to a model, before applying the algorithm to more complex models. The aim of this chapter is to research whether the combination of 1D hydrodynamic model of the estuary and Ensemble Kalman filter will perform better than only the 1D hydrodynamic model. To that end, the detailed setup is discussed in section 6.1, the performed experiments are discussed in section 6.2 and the results are shown in section 6.3.

6.1. Setup 1D model Western Scheldt estuary

The Western Scheldt is an estuary located in the southwest of the Netherlands. It flows from the North Sea border at Cadzand to the Dutch/Belgian border, see Figure 6.1 for the contour, close to the Belgian pilot of UNITED. Around the Western Scheldt, a few coastal cities are located where measurements of water characteristics are performed. These locations are Cadzand, Vlissingen, Terneuzen, Hansweert and Bath.

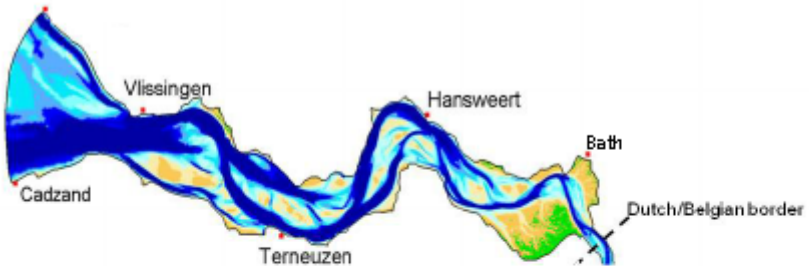


Figure 6.1: The Western scheldt contours and bathymetry (depth profile). Five coastal cities are considered, where measurement stations are present. Figure retrieved from [68] and adjusted.

The contour of the estuary can be approximated by a 1D grid of 100km long, see Figure 6.2, where the measurement stations are located at every 25th km, except for Bath. It is assumed that waterlevel h and water velocity u are measured at these stations, except for Bath where only h is measured. The observations are generated explained in section 5.2 with $\mathbf{R}_k = 0.01\mathbf{I}$.

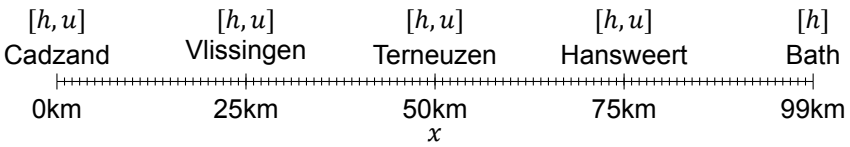


Figure 6.2: 1D grid approximation of the Western Scheldt estuary for the 1D model, with the five measuring stations Cadzand, Vlissingen, Terneuzen, Hansweert and Bath at respectively 0, 25, 50, 75 and 99km. It is assumed in each station observations of waterlevel h and/or velocity u are taken, displayed by the parameters above the measuring stations.

On this grid, the 1D hydrodynamic model with ensembles, discussed in algorithm 2 in subsection 3.4.1 is applied, as well as the 1D hydrodynamic model with Ensemble Kalman filter in algorithm 6. The model produces the time evolution of the state vector, containing waterlevel h and water velocity u. The settings used for both algorithms are given in Table 6.1. As boundary conditions, the waterlevel at Cadzand is retrieved from MATROOS series database [76] starting from 2013-12-05 00:00 to 2013-12-07 00:00 with a time interval of 10 min. The boundary at Bath is reflecting, thus the velocity will be 0.

Table 6.1: Settings used for the hydrodynamic model and hydrodynamic model with Ensemble Kalman filter. Some noise parameters are varied in experiments later.

Parameter	Value	Description
D	20 m	Depth
λ	$1.93 \cdot 10^{-4}$ s^{-1}	Friction coefficient
g	9.81 m s^{-2}	Acceleration of gravity
n	100	Number of grid cells
L	100 km	Length of the estuary
Δt	10 min	Time resolution
T	2 days	Total time of the model
t_0	2013-12-05 00:00	Starting time of the model
Δx	$\frac{L}{n+0.5}$km	Space resolution
N	50	Number of ensembles
α	$e^{\frac{-\Delta t}{6[\text{hours}]}} = e^{-\frac{1}{36}}$	Boundary noise coefficient
σ_N	0.2	Model error variance
$\mathbf{R}_k$	$0.01\mathbf{I}$	Observational error covariance
$\mathbf{W}_k$	$\begin{pmatrix} \mathbf{0} & \dots & \mathbf{0} \\ w_k & \dots & w_k \end{pmatrix}$	Model time-dependent noise
$\boldsymbol{\xi}_0$	$\mathbf{0}$	Initial ensemble

A run of the deterministic 1D model will result in estimates of the waterlevel h and water velocity u on the whole grid in Figure 6.2 from 2013-12-05 00:00 to 2 days later, see Figure 6.3. However, plotting on the whole grid as in Figure 6.3 is only possible for deterministic model runs, when there is no need for the plotting of error estimates. Therefore, estimates at separate locations are shown with error estimates. A run of the stochastic 1D model with $N = 50$ ensembles at Vlissingen is depicted in Figure 6.4 for waterlevel and 6.5 for water velocity. In these figures, clearly the periodic behavior of the tides is present in the estimates of the waterlevel and velocity. Furthermore, the standard deviation is around 0.2 for both, which was the set value.

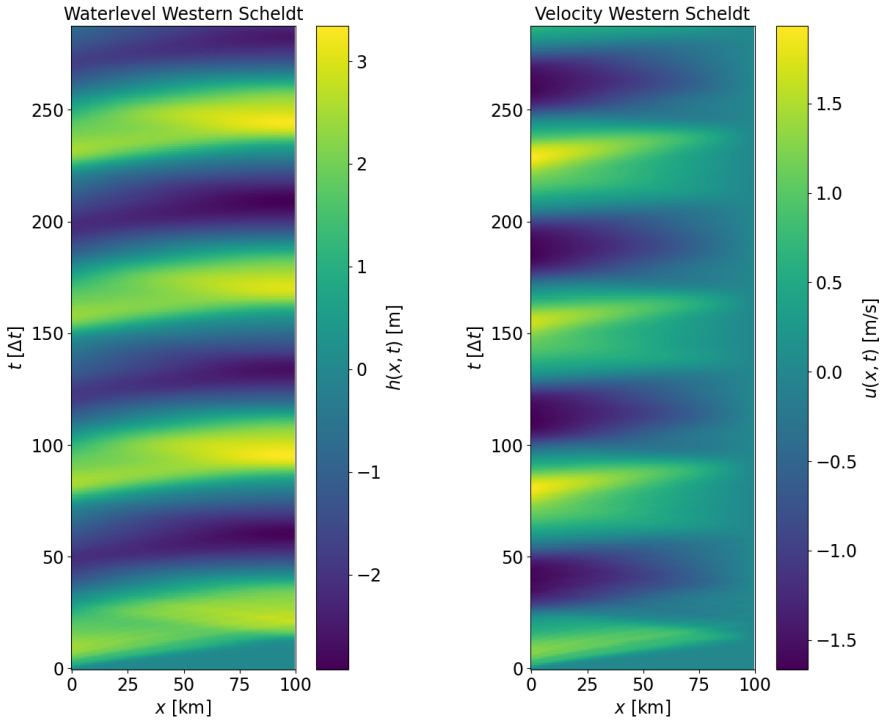


Figure 6.3: Deterministic estimates of the hydrodynamic model for the Western Scheldt determined via algorithm 2 in subsection 3.4.1, plotted as a function of position x in the Western Scheldt and time Δt after t_0 = 2013-12-05 00:00. Left: Waterlevel $h(x,t)$. Right: Water velocity $u(x,t)$.

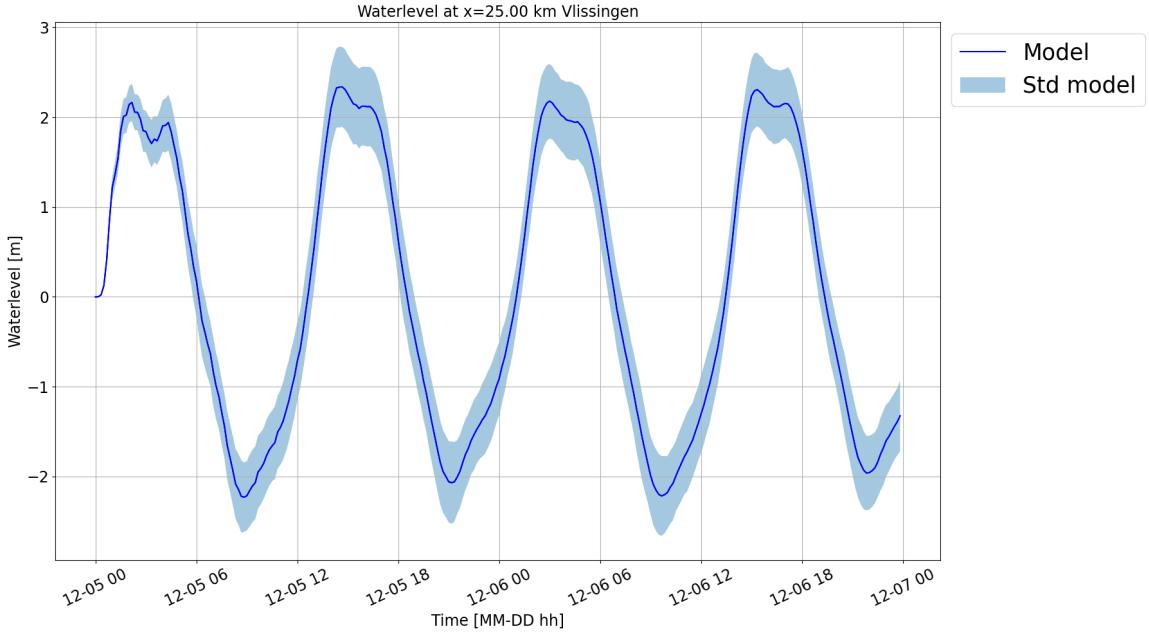


Figure 6.4: Waterlevel estimate with standard deviation at Vlissingen, modelled stochastically with algorithm 2 with settings in Table 6.1 over 2 days from 2013-12-05 00:00 for $N = 50$.

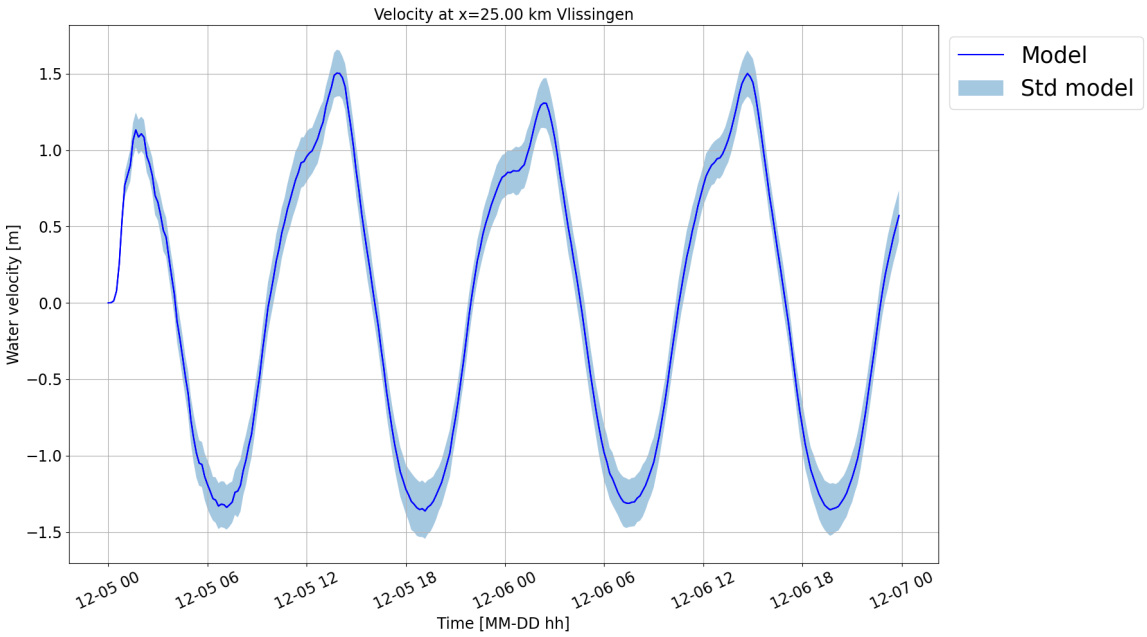


Figure 6.5: Velocity estimate with standard deviation at Vlissingen, modelled stochastically with algorithm 2 with settings in Table 6.1 over 2 days from 2013-12-05 00:00 for $N = 50$.

6.2. Experiment setup

Since the output of the Ensemble Kalman filter filter depends on the input settings, experiments are performed to test the Ensemble Kalman filter (algorithm 6 in section 5.1) in comparison to the model (algorithm 2 in subsection 3.4.1). In all experiments mentioned below, the true state and observations are generated. The true state is generated by taking one of the estimates from the ensemble model run. In Figure 6.6 and 6.7, the waterlevel and velocity predictions of the model at Vlissingen are shown with standard deviation and chosen true state. The observations are generated from this true state with observations error covariance $\mathbf{R}_k$, unless mentioned otherwise.

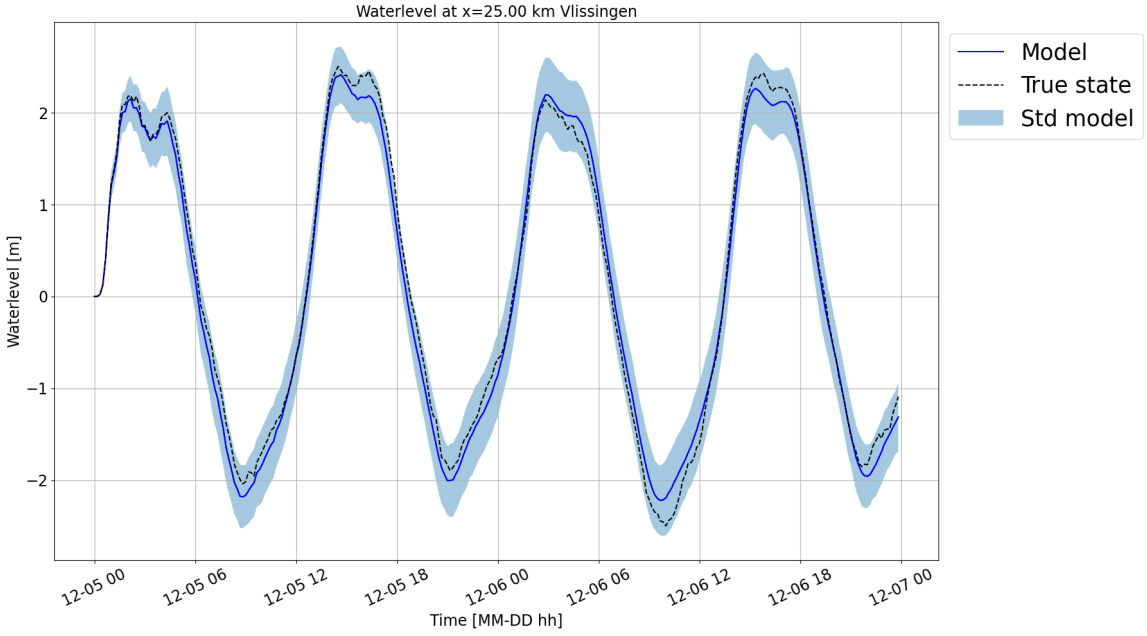


Figure 6.6: The waterlevel prediction with standard deviation at Vlissingen, modelled stochastically with algorithm 2 with settings in Table 6.1, and chosen true state from the ensembles, over two days from 2013-12-05 00:00 until 2013-12-07 00:00.

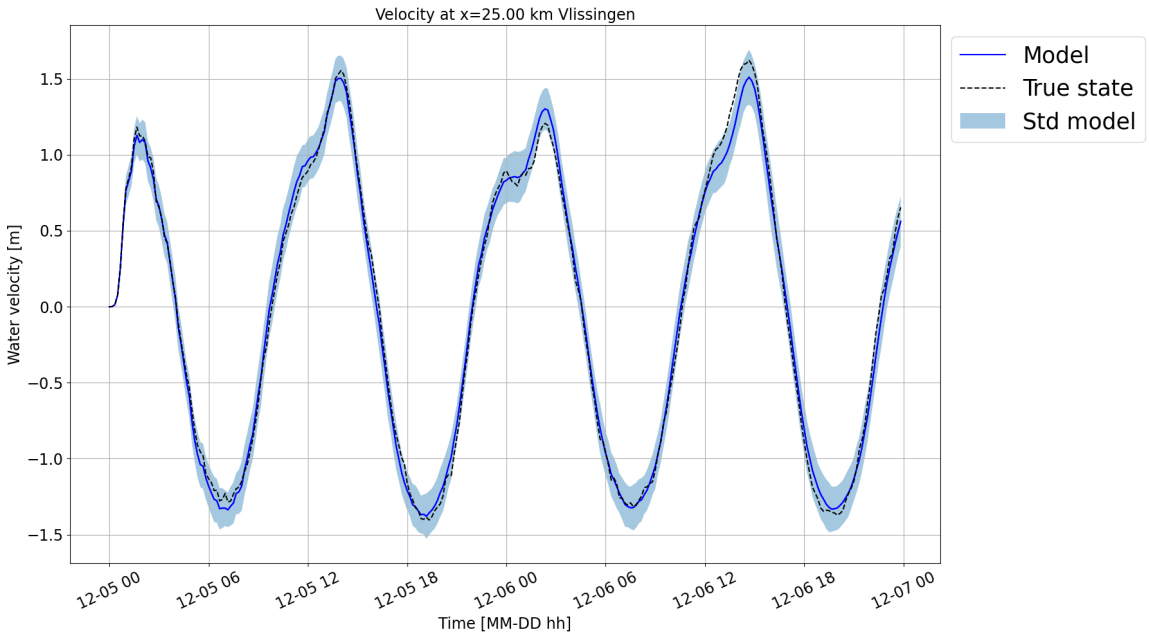


Figure 6.7: The water velocity prediction with standard deviation at Vlissingen, modelled stochastically with algorithm 2 with settings in Table 6.1, and chosen true state from the ensembles, over two days from 2013-12-05 00:00 until 2013-12-07 00:00.

The experiments to test the Ensemble Kalman filter (EnKF) are mentioned here:

1. **Test implementation model with EnKF in algorithm 6 in section 5.1**
   At first, it is checked if algorithm 6 in section 5.1 with settings in Table 6.1 is implemented correctly in Python. To this end, the algorithm will assimilate waterlevel values at each grid location, called EnKF5, see Figure 6.8. One way to check the performance is by checking if the rmse between model with EnKF (EnKF5) and the true state is equal to the standard deviation of EnKF5. Furthermore, when the rmse for the model is larger than for EnKF5, it means the filter has a smaller error and thus has a better estimate.

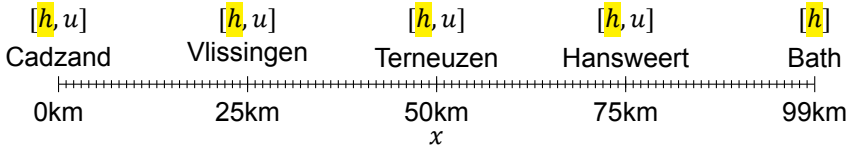


Figure 6.8: EnKF5. Assimilation of only waterlevel at each location (yellow marked).

2. **Effect of more assimilated physical variables**
   To be able to see if assimilation of more observations of another physical variable has an effect on the estimate, the rmse for algorithm 6 with only waterlevel assimilated (EnKF5) and algorithm 6 with both waterlevel and water velocity assimilated are compared, called EnKF9, see Figure 6.9.

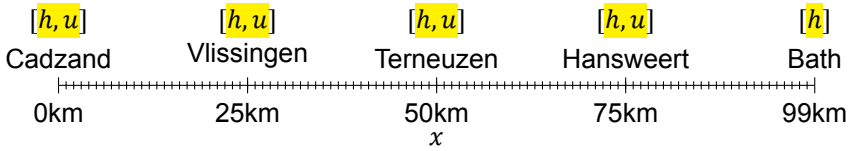


Figure 6.9: EnKF9. Assimilation of waterlevel and water velocity at each location (yellow marked).

3. **Effect of assimilation location**
   This experiment will test the effect of assimilation at different locations on the total errors of the estimate. To that end, only one physical variable per location is assimilated, for example waterlevel

at only Cadzand, see Figure 6.10. Furthermore, three cases with assimilation from waterlevel observations at Vlissingen and Terneuzen (h Vli h Ter), assimilation from velocity observations at Vlissingen and Terneuzen (u Vli u Ter) and assimilation from waterlevel and velocity observations at Vlissingen (h Vli u Vli) are investigated. These will show the importance of waterlevel observations to velocity observations, and which is most useful to assimilated. Furthermore, the effect from assimilation on Cadzand on other measuring stations can be seen by comparing the rmse of the estimates in each measuring station.

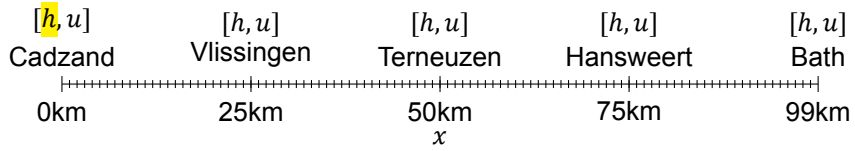


Figure 6.10: EnKF h Cad. Assimilation of only waterlevel at Cadzand (yellow marked). For other physical variables and other measuring stations, the EnKF estimate is similarly named.

4. **Effect of frequency of assimilation**
   In this experiment, it is examined if the frequency of assimilation has an effect on the errors of the predictions. To that end, the frequency of assimilation is changed from every time step to only once in every 40 time steps. The experiment is performed on EnKF5.

5. **Effect of observation error covariance $\mathbf{R}_k$**
   In this experiment, the observation error covariance $\mathbf{R}_k$ is varied, and thus the error on the true state and observations, to see the effect of the trade-off of the Kalman Gain between the sample error covariance $\mathbf{P}_k^a$ and the observation error covariance. To that end, $\mathbf{R}_k$ is varied from $0.01\mathbf{I}$ to $5\mathbf{I}$. To that end, the true state and observations are varied. In comparison, the model error covariance is determined by the boundary noise coefficient α, recall Equation 3.21 in section 3.4, which is taken to be around 0.2. Therefore, the range of values for $\mathbf{R}_k$ has values smaller and larger than α, and the effect of the Kalman Gain is investigated.

6. **Effect of other observations**
   How well does the EnKF5 perform while assimilating with observations not generated from the model? In this experiment, it is assumed that the true state is more distinct from the model. It is investigated if the EnKF is able to create a good prediction while assimilating with observations generated from this true state. To that end, three cases are distinguished:

   - Observations from the hydrodynamic model with offset of 50 cm for waterlevel and 10 cm/s for velocity.
   - Observations from the hydrodynamic model with delay in time of $5\Delta t$.
   - Observations created with a sin function not close to the model, defined as: $\boldsymbol{z}_k = \mathbf{sin}(\phi k)$ with approximately the same period as the model, defined as $\phi = \frac{2\pi}{45000}$.

6.3. Results 1D experiments

In this section, the 6 experiments mentionend in section 6.2 are executed and the results shown and examined. To that end, first the results of the estimates of the EnKF5, Figure 6.11 and 6.12, and EnKF9, Figure 6.13 and 6.14 are shown. In red, the estimates of the EnKF are displayed with standard deviation in yellow. The model estimates are shown in blue with lightblue standard deviation and the true state and measurements are shown with black striped and dots respectively. Notice from these figures that both EnKF5 and EnKF9 have estimate close to the true state, closer than only the hydrodynamic model. Furthermore, the standard deviation for both EnKF models decreased, which is as expected, since the EnKF algorithm produces lower standard deviation.

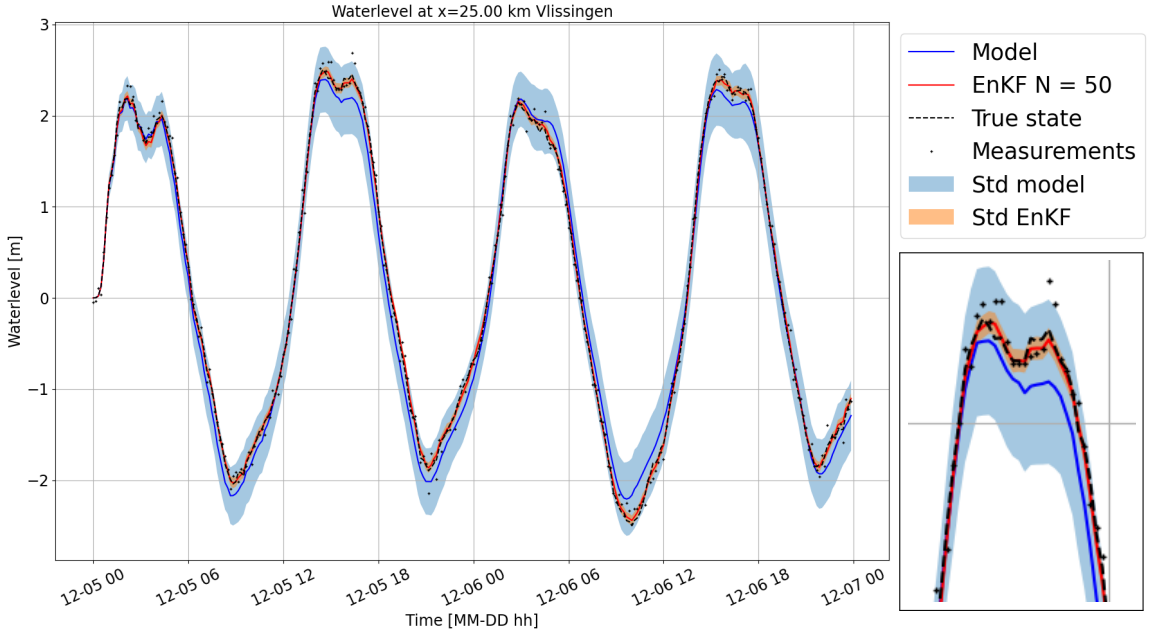


Figure 6.11: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF5 with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model. Black dots: measurements.

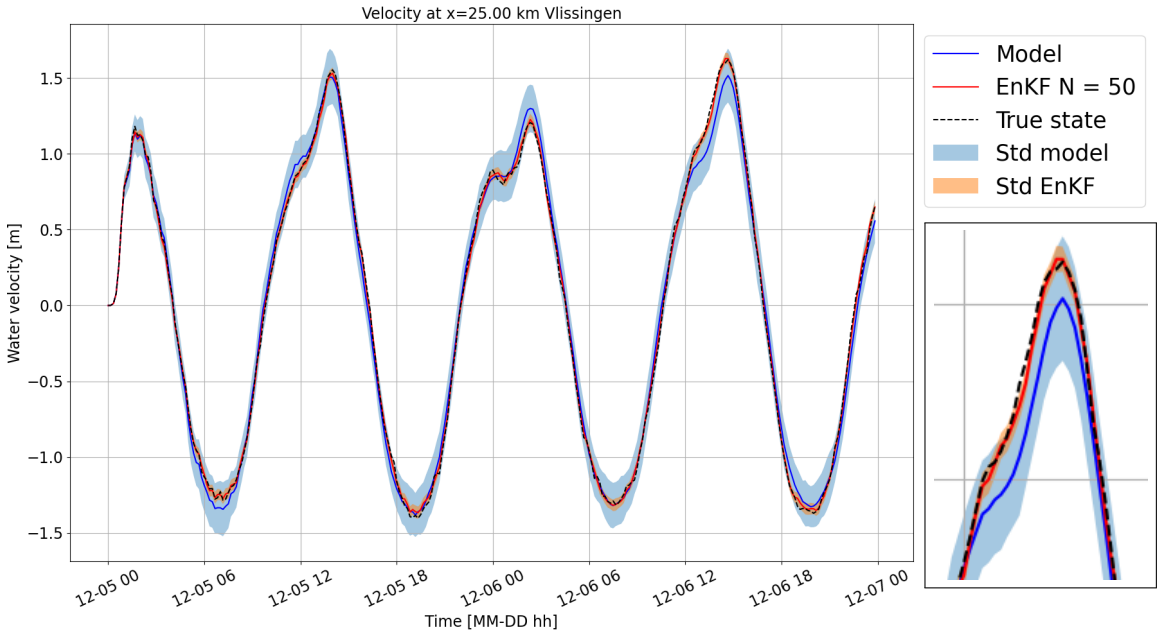


Figure 6.12: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF5 with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model.

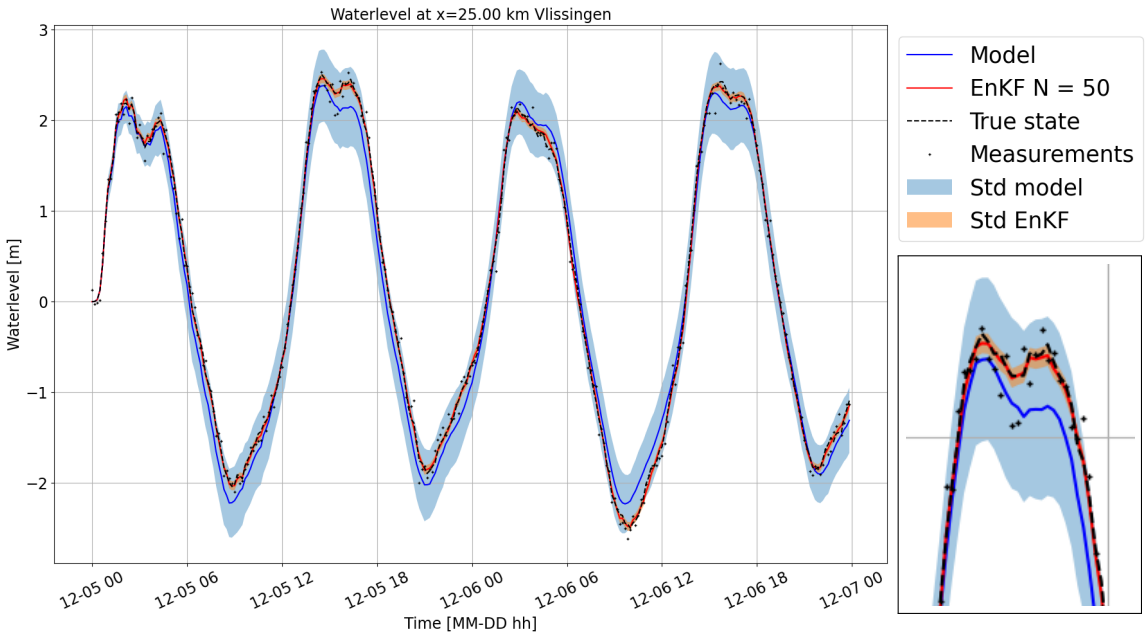


Figure 6.13: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF9 with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model prediction. Red: Estimate of EnKF9. Yellow: standard deviation of EnKF9. Black striped: chosen true state from the model. Black dots: measurements.

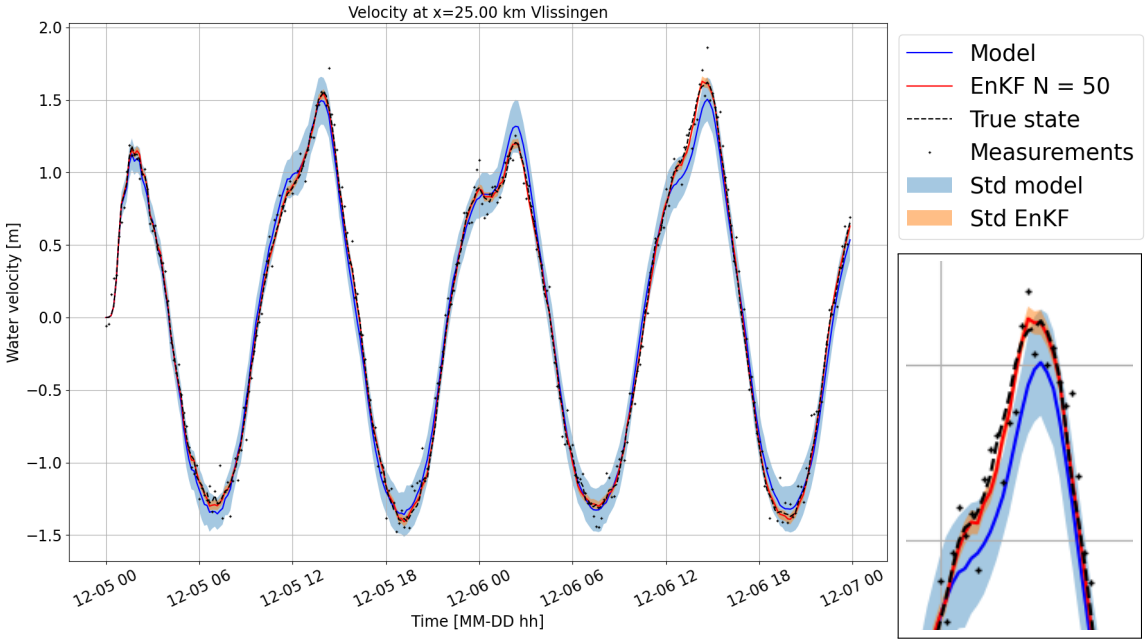


Figure 6.14: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF9 with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF9. Yellow: standard deviation of EnKF9. Black striped: chosen true state from the model. Black dots: measurements.

6.3.1. Test implementation EnKF

To test the implementation of EnKF5, the total std and rmse are computed at each measuring station using Equation 5.6 and Equation 5.4 in section 5.3, see Figure 6.15. This figure shows that the rmse

and std for the EnKF model are similar in value. Therefore, the EnKF filter is applied correctly on this model. The spread in prediction for the EnKF5 model is clearly lower than for the model without EnKF5, as can be seen by comparing the std EnKF5 with the std model for each measuring station. Furthermore, the rmse for EnKF5 decreases with almost a factor 4 for waterlevel and 2,5 for water velocity, which indicates that applying EnKF5 indeed results into a more precise model.

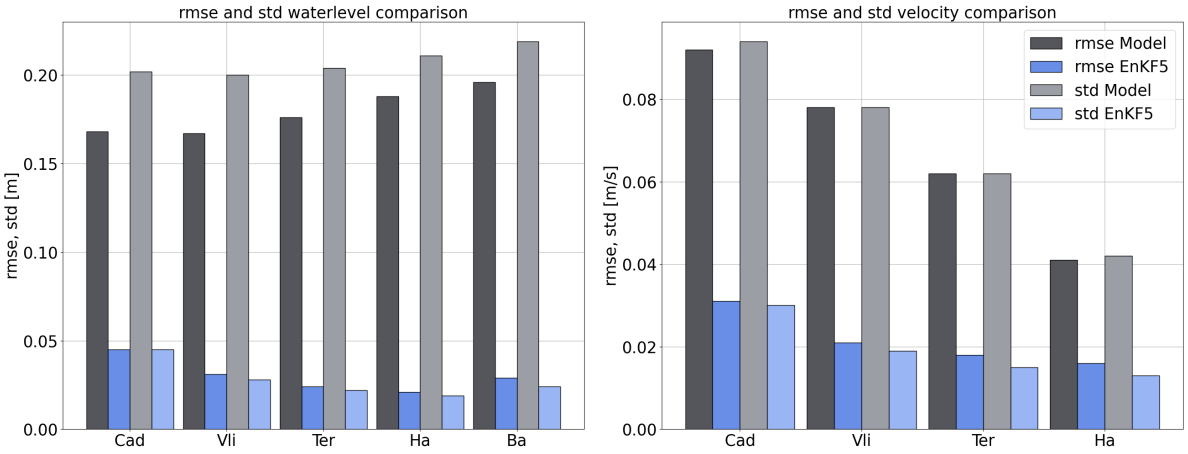


Figure 6.15: Left: rsme and std of the waterlevel estimate of model (in black/grey) and EnKF5 (in blue) for each measuring station. Right: rmse and std of the velocity estimate of model (in black/grey) and EnKF5 (in blue) for each measuring station.

6.3.2. Effect of more assimilated physical variables

To be able to see if the assimilation of more observations of another physical variable has an effect on the prediction, the rmse for EnKF5 with only waterlevel assimilated and EnKF9 with both waterlevel and water velocity assimilated are compared. The rmse is shown in Figure 6.16 and the standard deviations in Figure 6.17. When extra assimilating the velocity observations in four extra measuring

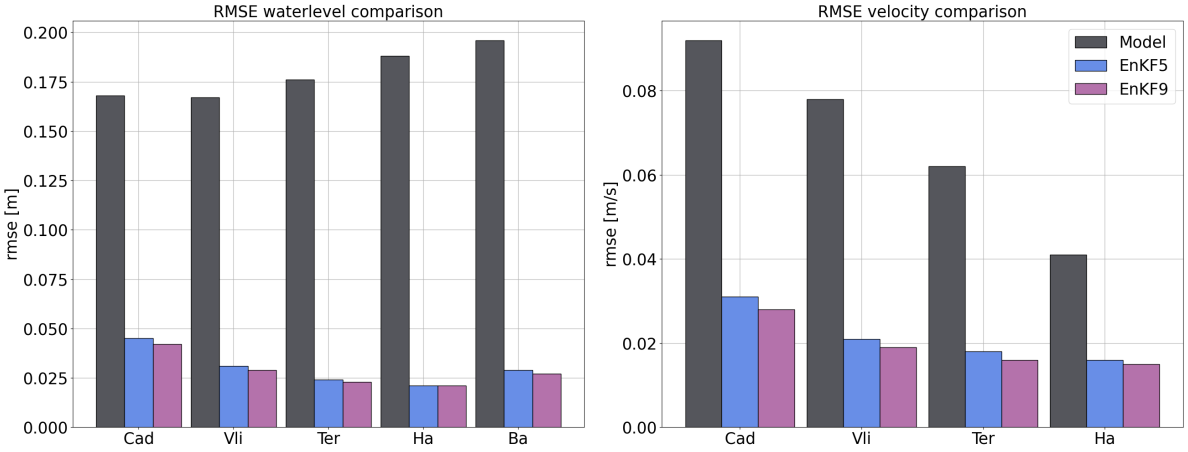


Figure 6.16: Left: rsme of the waterlevel estimate of model (in grey), EnKF5 (in blue) and EnKF9 (in purple) for each measuring station. Right: rmse of the velocity estimate of model (in grey), EnKF5 (in blue) and EnKF9 (in purple) for each measuring station.

stations for EnKF9, both the rmse and std values decrease slightly in comparison to EnKF5, but are still much lower than only the model rmse. Therefore, when more variables are assimilated, the estimate becomes slightly more accurate. To conclude, when extra observations at different measuring stations become available, assimilation of those will result in a slightly more accurate estimate.

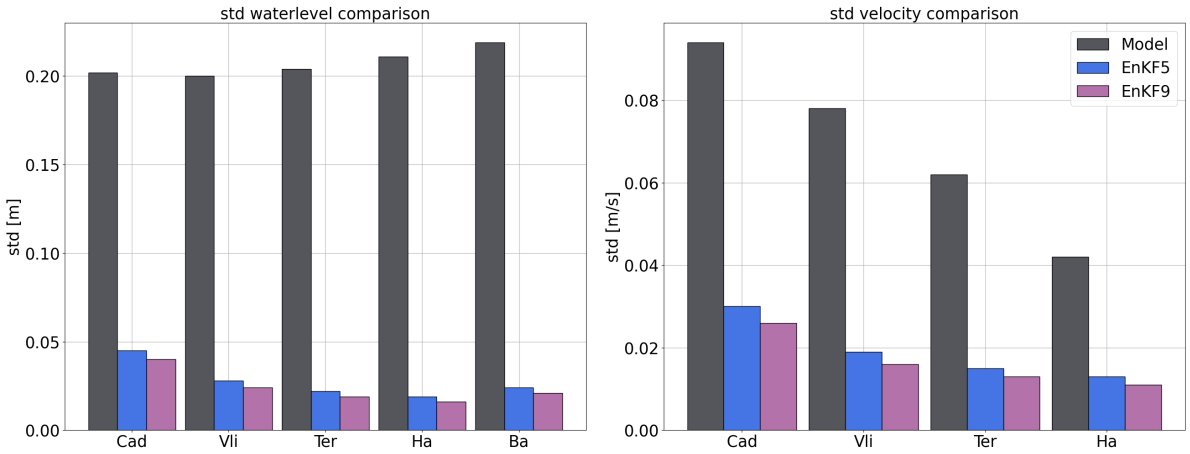


Figure 6.17: Left: std of the waterlevel estimate of model (in grey), EnKF5 (in blue) and EnKF9 (in purple) for each measuring station. Right: std of the velocity estimate of model (in grey), EnKF5 (in blue) and EnKF9 (in purple) for each measuring station.

6.3.3. Effect of assimilation location

To see the effect of assimilation at different locations on the total errors of the estimate, only one physical variable per location is assimilated, for example waterlevel at only Cadzand (EnKF h Cad), in (1)-(9). The estimates are not shown here, since displaying all at all measuring stations would result in 120 more or less the same figures, however some are shown in Appendix B. The rmse of all EnKF models at all measuring stations are plotted for waterlevel in Figure 6.18 and for velocity in Figure 6.19. Furthermore, the cases with two assimilated observations EnKF hVli hTer, EnKF uVli uTer and EnKF hVli uVli ((10)-(12)) are shown as well, to see if assimilation of waterlevel or of velocity is preferred for a better estimate.

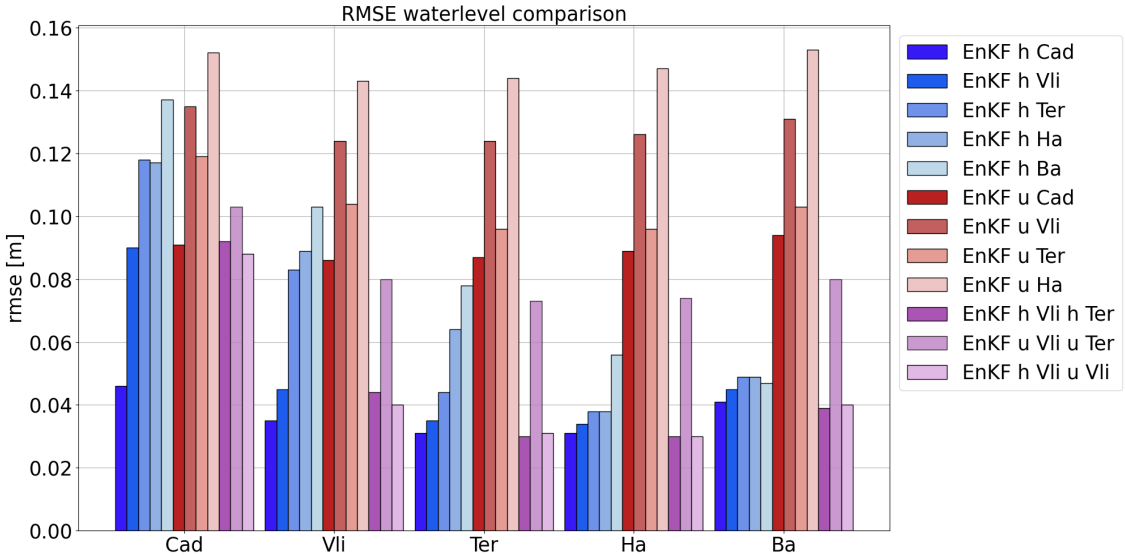


Figure 6.18: rmse in waterlevel h estimate per measuring station per model. The models are from left to right respectively are EnKF assimilated with (1) EnKF with observations of h Cadzand, (2) EnKF with observations of h Vlissingen, (3) EnKF with observations of h Terneuzen, (4) EnKF with observations of h Hansweert, (5) EnKF with observations of h Bath, (6) EnKF with observations of u Cadzand, (7) EnKF with observations of u Vlissingen, (8) EnKF with observations of u Terneuzen, (9) EnKF with observations of u Hansweert, (10) EnKF with observations of h Vlissingen and h Terneuzen, (11) EnKF with observations of u Vlissingen and u Terneuzen, (12) EnKF with observations of h Vlissingen and u Vlissingen.

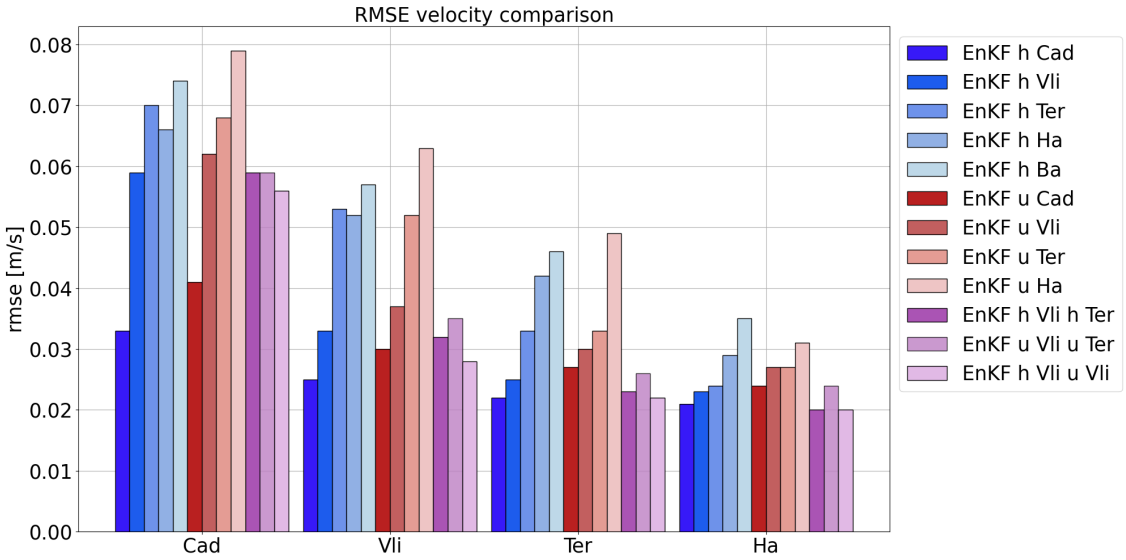


Figure 6.19: rmse in velocity u estimate per measuring station per model. The models are from left to right respectively are EnKF assimilated with (1) EnKF with observations of h Cadzand, (2) EnKF with observations of h Vlissingen, (3) EnKF with observations of h Terneuzen, (4) EnKF with observations of h Hansweert, (5) EnKF with observations of h Bath, (6) EnKF with observations of u Cadzand, (7) EnKF with observations of u Vlissingen, (8) EnKF with observations of u Terneuzen, (9) EnKF with observations of u Hansweert, (10) EnKF with observations of h Vlissingen and h Terneuzen, (11) EnKF with observations of u Vlissingen and u Terneuzen, (12) EnKF with observations of h Vlissingen and u Vlissingen.

Looking at the rmse values in both Figure 6.18 in 6.19, notice that upon assimilating observations in measuring station Cadzand, the rmse is lowest. Furthermore, it seems that each measuring station farther from the Cadzand will have higher rmse. Therefore, assimilation at Cadzand is most important. An hypothesis is that assimilation on the boundary where the noise is applied is the most useful. However, it is beyond the scope of this thesis to prove this. Furthermore, it is noticed that most EnKFs in blue, where waterlevel is assimilated, have lower rmse than their velocity assimilated (in red) counter parts at the same measuring station. Also for the assimilation with two physical parameters in purple, the assimilation where waterlevel is incorporated is lower than without assimilation of waterlevel. To conclude, assimilation of waterlevel observations results in a better estimate than assimilation of water velocity. Even more, when comparing (10) and (12) in purple again, it is noticed that the case (12) has slightly lower error in measuring stations than case (10). Therefore, for a slightly better estimate, it is preferred to have observations of multiple physical variables compared to only one physical variables.

6.3.4. Effect of frequency of assimilation

In this experiment, it is examined if the frequency of assimilation has an effect on the errors of the estimates. Therefore, the frequency of assimilation is changed from every time step up until only once in every 40 time steps. The experiment is performed on EnKF5. Some of the estimates for $T = 2\Delta t$, $T = 10\Delta t$ and $T = 40\Delta t$ at Vlissingen are shown in Appendix C. The result of the rmse can be seen in Figure 6.20 for waterlevel and 6.21 for velocity. Also the rmse of the model without EnKF is shown.

From both plots, a clear trend is seen. When increasing the assimilation frequency, the rmse increases as well. Important to notice is that even with assimilation frequency $T = 40\Delta t$, the rmse in all cases is lower than only the model rmse. Therefore, applying the EnKF with sparse observations in time still results in a better estimate than in the case that no EnKF is applied. It can be argued that if taking many observations in time is costly for measuring stations, the assimilation frequency and thus frequency of taking observations may be reduced to a value of rmse what is desired.

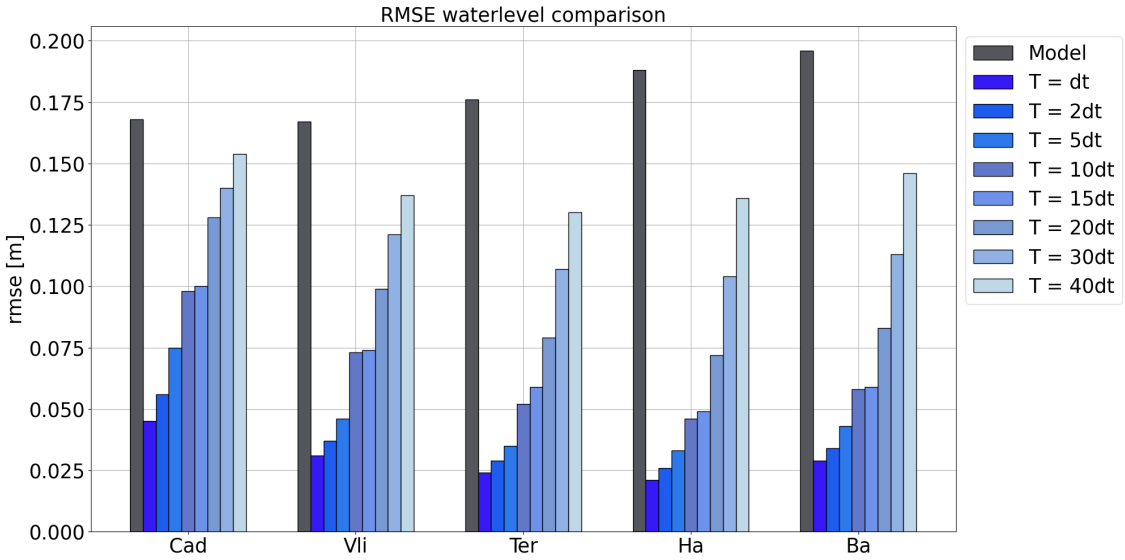


Figure 6.20: rmse in waterlevel h estimate per measuring station per model. The models from left to right respectively are (1) Model, (2) EnKF with $T = \Delta t$, (3) EnKF with $T = 2\Delta t$, (4) EnKF with $T = 5\Delta t$, (5) EnKF with $T = 10\Delta t$, (6) EnKF with $T = 15\Delta t$, (7) EnKF with $T = 20\Delta t$, (8) EnKF with $T = 30\Delta t$ and (9) EnKF with $T = 40\Delta t$.

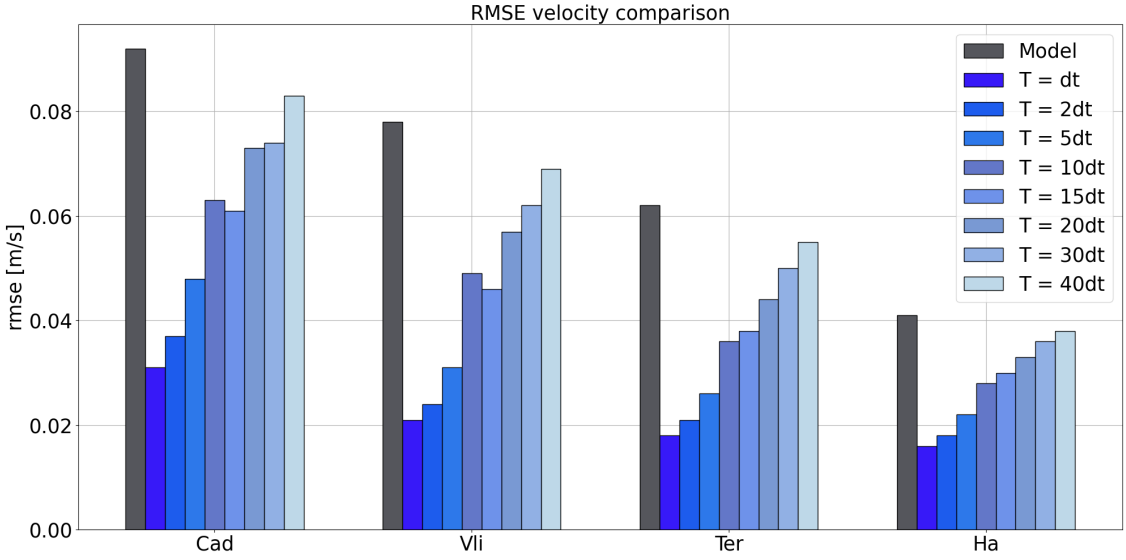


Figure 6.21: rmse in velocity u estimate per measuring station per model. The models are from left to right respectively are (1) Model, (2) EnKF with $T = \Delta t$, (3) EnKF with $T = 2\Delta t$, (4) EnKF with $T = 5\Delta t$, (5) EnKF with $T = 10\Delta t$, (6) EnKF with $T = 15\Delta t$, (7) EnKF with $T = 20\Delta t$, (8) EnKF with $T = 30\Delta t$ and (9) EnKF with $T = 40\Delta t$.

6.3.5. Effect of observation error covariance

In this experiment, the observation error covariance $\mathbf{R}_k$ is varied, to see the effect of the trade-off of the Kalman Gain between the sample error covariance $\mathbf{P}_k^a$ and the observation error covariance discussed in section 4.2. $\mathbf{R}_k$ is varied from $0.01\mathbf{I}$ to $5\mathbf{I}$. In comparison, the model error covariance is determined by the boundary noise coefficient α, which is taken to be around 0.2. Therefore, $\mathbf{R}_k$ takes on values smaller and larger than α, and thus the Kalman Gain will favor the observations for lower $\mathbf{R}_k$ and favor the model for higher $\mathbf{R}_k$.

The estimates are not shown due to their similarities with previous estimates. In Figure 6.22 and 6.23 the rmse of these models is shown. A clear trend is visible, when the observation error covariance $\mathbf{R}_k$ becomes large, the rmse becomes large as well for every measuring station and both waterlevel and velocity. Notice that $\mathbf{R}_k = 5\mathbf{I}$ results in a quite large observation error, but that in this case still the

model with EnKF will outperform the model without EnKF (in black).

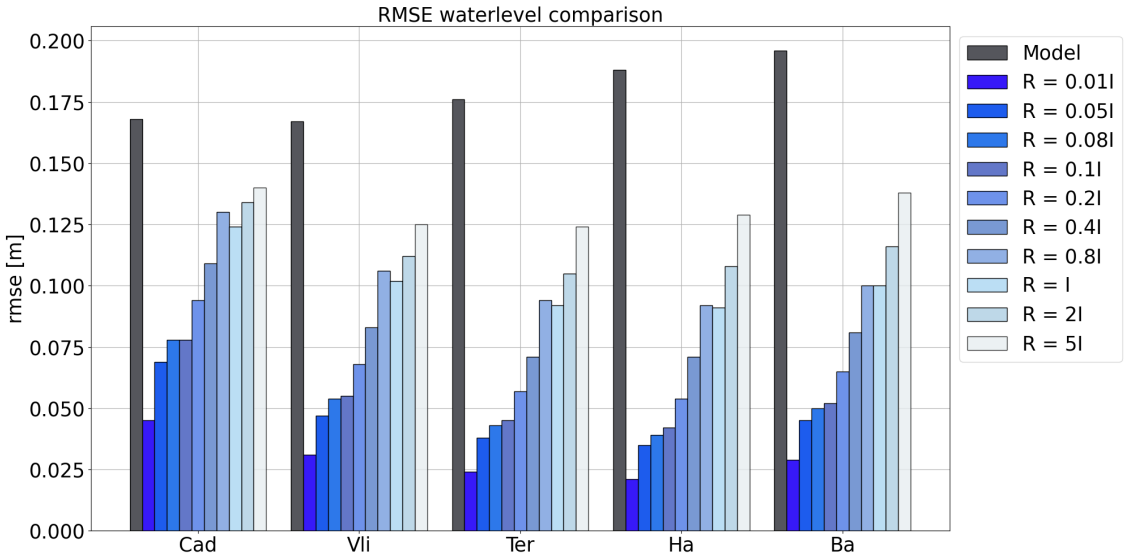


Figure 6.22: rmse in waterlevel h estimate per measuring station per model. The models are from left to right respectively are (1) Model, (2) EnKF with $\mathbf{R}_k = 0.01\mathbf{I}$, (3) EnKF with $\mathbf{R}_k = 0.05\mathbf{I}$, (4) EnKF with $\mathbf{R}_k = 0.08\mathbf{I}$, (5) EnKF with $\mathbf{R}_k = 0.1\mathbf{I}$, (6) EnKF with $\mathbf{R}_k = 0.2\mathbf{I}$, (7) EnKF with $\mathbf{R}_k = 0.4\mathbf{I}$, (8) EnKF with $\mathbf{R}_k = 0.8\mathbf{I}$, (9) EnKF with $\mathbf{R}_k = \mathbf{I}$, (10) EnKF with $\mathbf{R}_k = 2\mathbf{I}$,(11) EnKF with $\mathbf{R}_k = 5\mathbf{I}$.

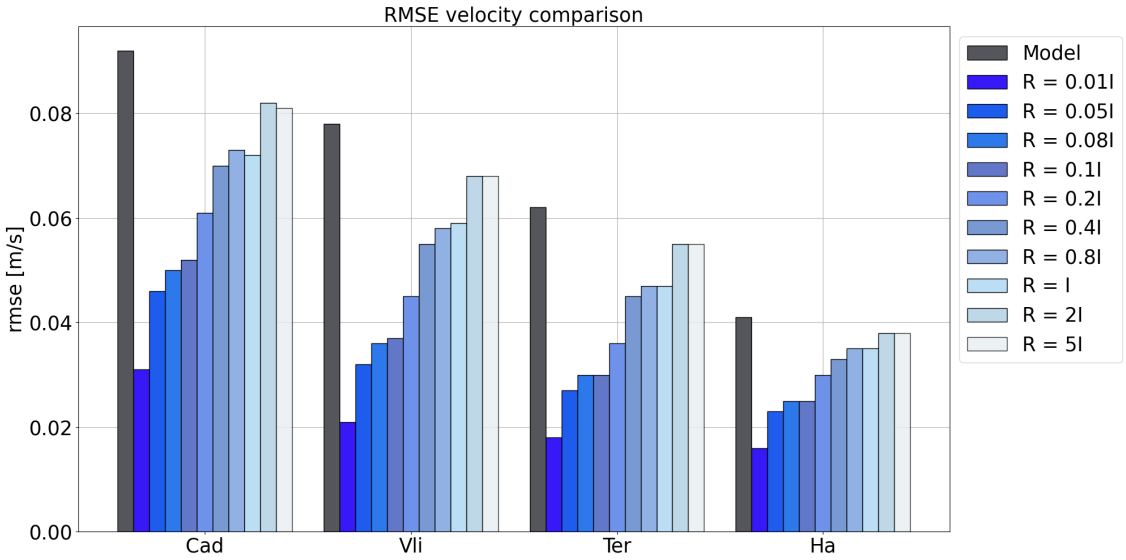


Figure 6.23: rmse in water velocity u estimate per measuring station per model. The models are from left to right respectively are (1) Model, (2) EnKF with $\mathbf{R}_k = 0.01\mathbf{I}$, (3) EnKF with $\mathbf{R}_k = 0.05\mathbf{I}$, (4) EnKF with $\mathbf{R}_k = 0.08\mathbf{I}$, (5) EnKF with $\mathbf{R}_k = 0.1\mathbf{I}$, (6) EnKF with $\mathbf{R}_k = 0.2\mathbf{I}$, (7) EnKF with $\mathbf{R}_k = 0.4\mathbf{I}$, (8) EnKF with $\mathbf{R}_k = 0.8\mathbf{I}$, (9) EnKF with $\mathbf{R}_k = \mathbf{I}$, (10) EnKF with $\mathbf{R}_k = 2\mathbf{I}$,(11) EnKF with $\mathbf{R}_k = 5\mathbf{I}$.

6.3.6. Effect of other generated observations

It is investigated if the EnKF5 is able to create a good estimate while assimilating with observations more distinct from the model. To that end, three cases are distinguished:

- Observations from the hydrodynamic model with offset of 50 cm for waterlevel and 10 cm/s for velocity.
- Observations from the hydrodynamic model with delay in time of $5\Delta t$.

- Observations created with a sin function not close to the model, defined as:

$$\boldsymbol{z}_k = \mathbf{sin}(\phi k), \tag{6.1}$$

with approximately the same period as the model, defined as $\phi = \frac{2\pi}{45000}$.

For these experiments, the estimates are shown, since these vary from previous estimate plots.

Offset
The estimates at Vlissingen for the models with offset in observations are shown in Figure 6.25 and 6.26. The true state in black stripes is not located within the standard deviation of the model estimate anymore. However, when applying the EnKF5, the estimate of EnKF5 in red is very close to the true state, for both waterlevel and velocity. Therefore, the EnKF5 is still able to create a good estimate when having measurements varying slightly from the model.

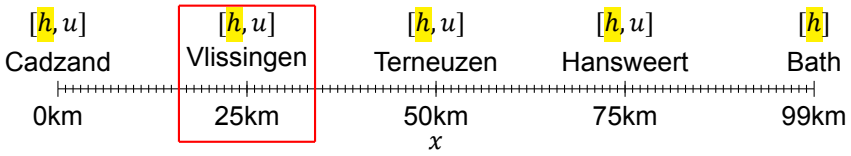


Figure 6.24: EnKF5. Assimilation of only waterlevel at each location (yellow marked). The red box markes that estimate of Figure 6.25 and 6.26 are shown at Vlissingen.

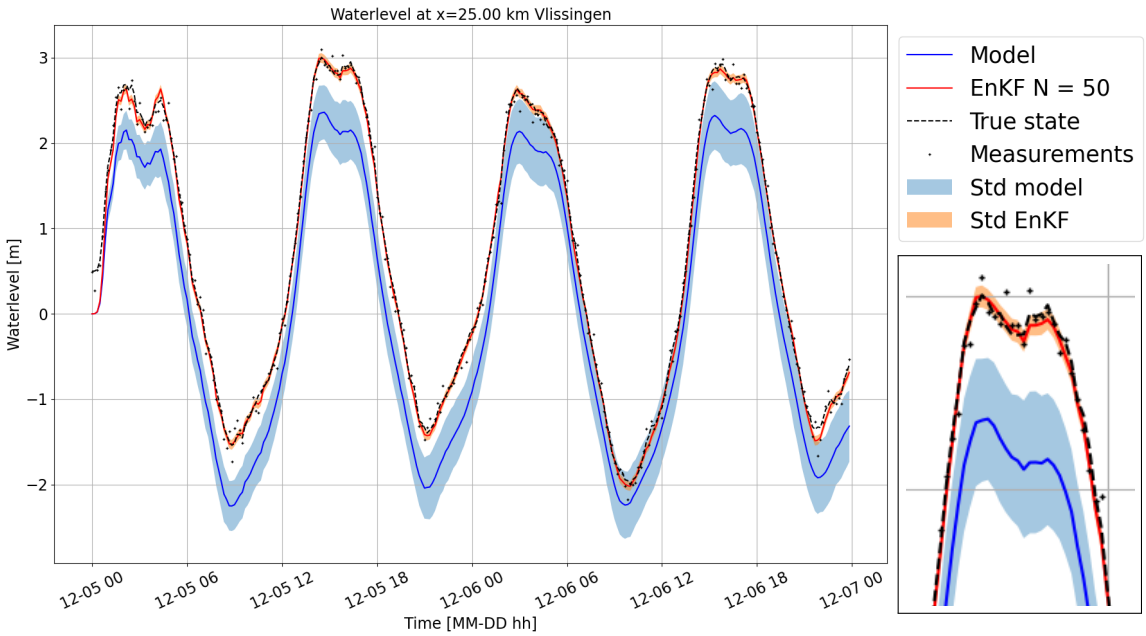


Figure 6.25: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF5 with generated measurements with offset of 50 cm for waterlevel and 10 cm/s for velocity from the model and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model. Black dots: generated measurements.

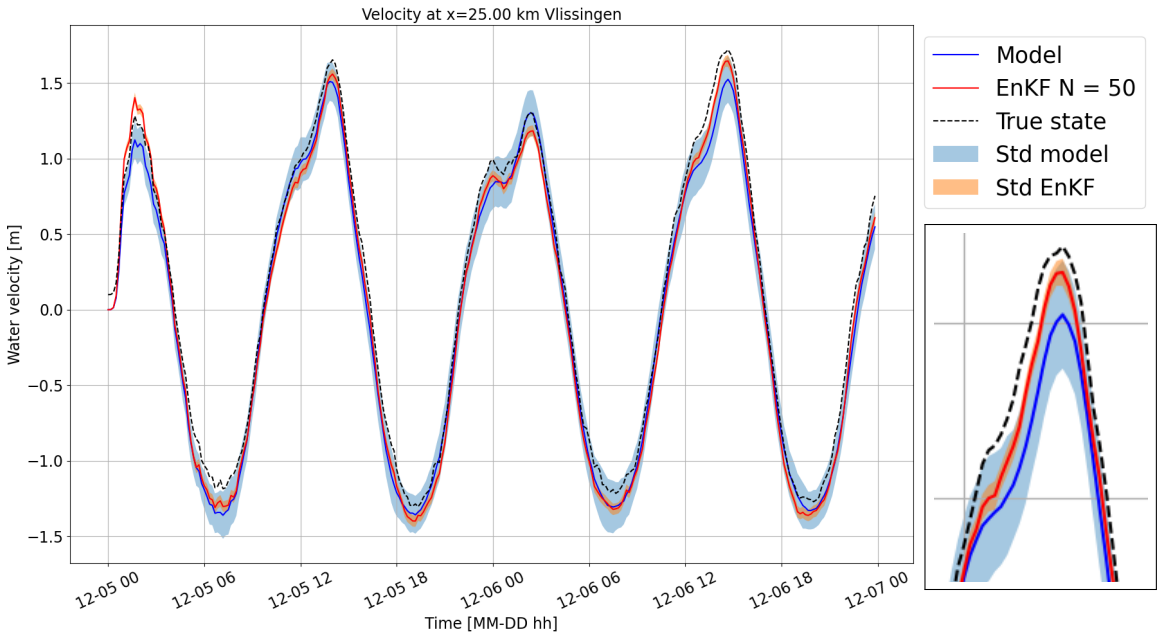


Figure 6.26: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF5 with generated measurements with offset of 50 cm for waterlevel and 10cm/s for velocity from the model and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model.

Delay

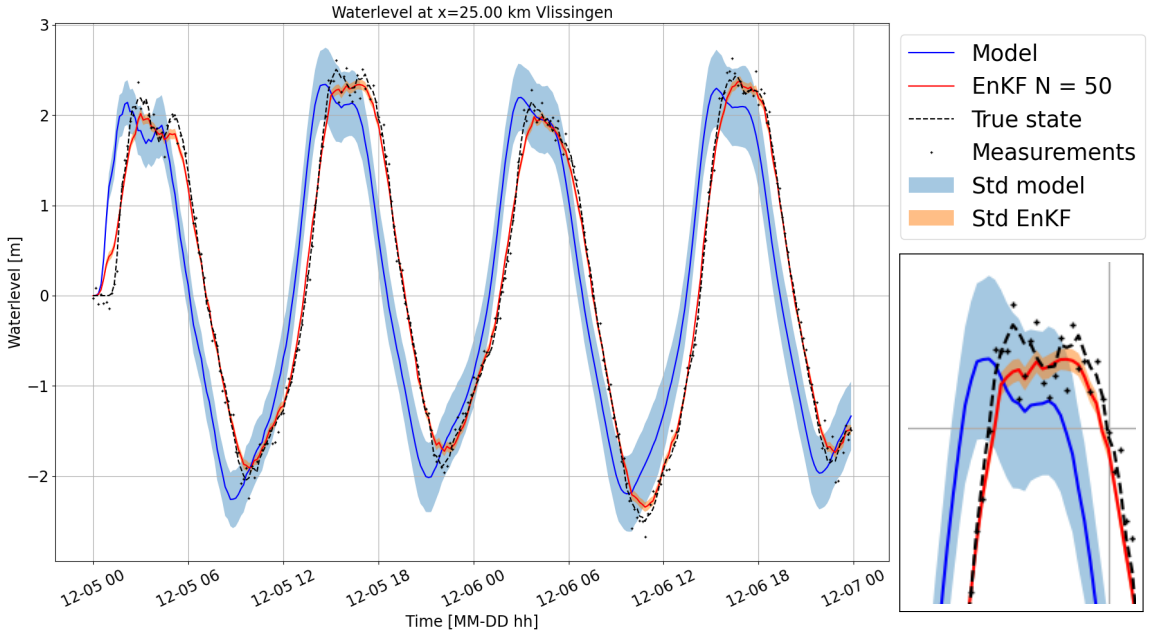


Figure 6.27: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF5 with generated measurements with delay of $5\Delta t$ from the model and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model. Black dots: generated measurements.

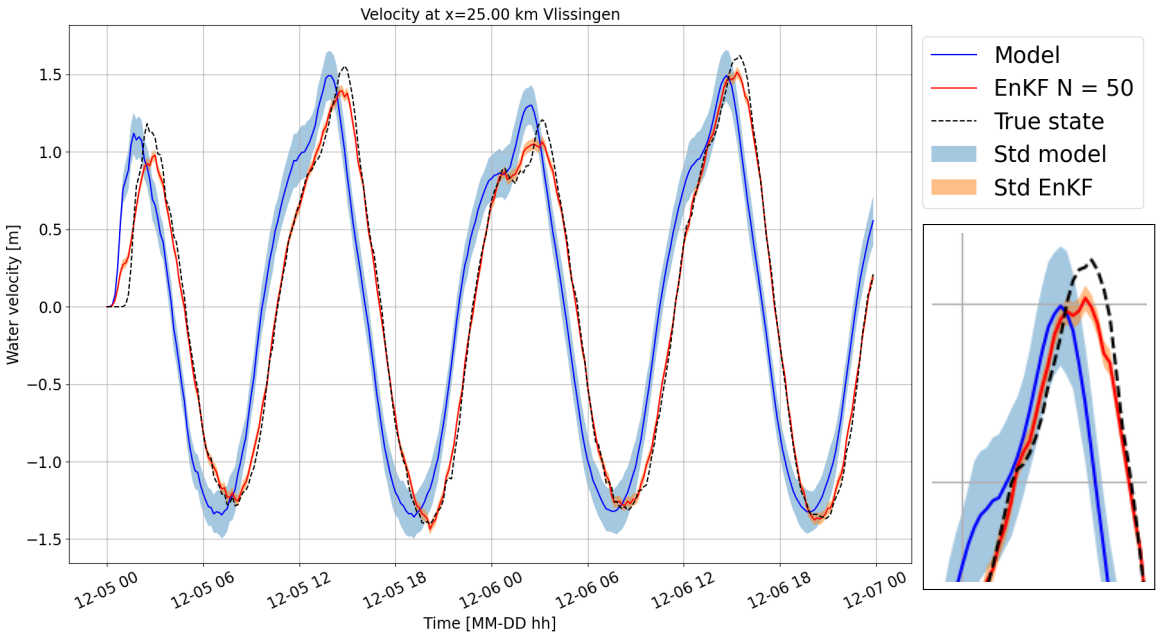


Figure 6.28: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF5 with generated measurements with delay of $5\Delta t$ from the model and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model.

The estimates at Vlissingen for the models with delay of $5\Delta t$ in observations are shown in Figure 6.27 and 6.28. The true state in black stripes is not located within the standard deviation of the model estimate. When applying the EnKF5, the estimate of EnKF5 in red is very close to the true state, for both waterlevel and velocity. Therefore, the EnKF5 is still able to create a good estimate when having measurements with a slight time delay.

Sin

The estimates at Hansweert (Figure 6.29) for the models with EnKF with assimilated observations generated by Equation 6.1 are shown in Figure 6.30 and 6.31. The true state in black stripes is not located within the standard deviation of the model estimate. When applying the EnKF5, the estimate of EnKF5 in red is somewhat close to the true state for waterlevel, however for velocity deviates very much. Notice that velocity is not assimilated in this EnKF model. The EnKF tries to account for the physics of the model for the velocity prediction, but does not get close to the true state. It has to be noted that these observations could be invalid observations with respect to the hydrodynamic model. To conclude, the EnKF cannot predict the chosen true state very well in this situation.

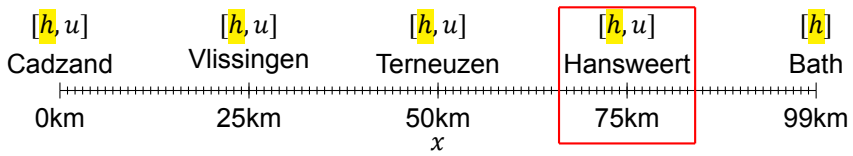


Figure 6.29: EnKF5. Assimilation of only waterlevel at each location (yellow marked). The red box markes that estimates of Figure 6.25 and 6.26 are shown at Hansweert.

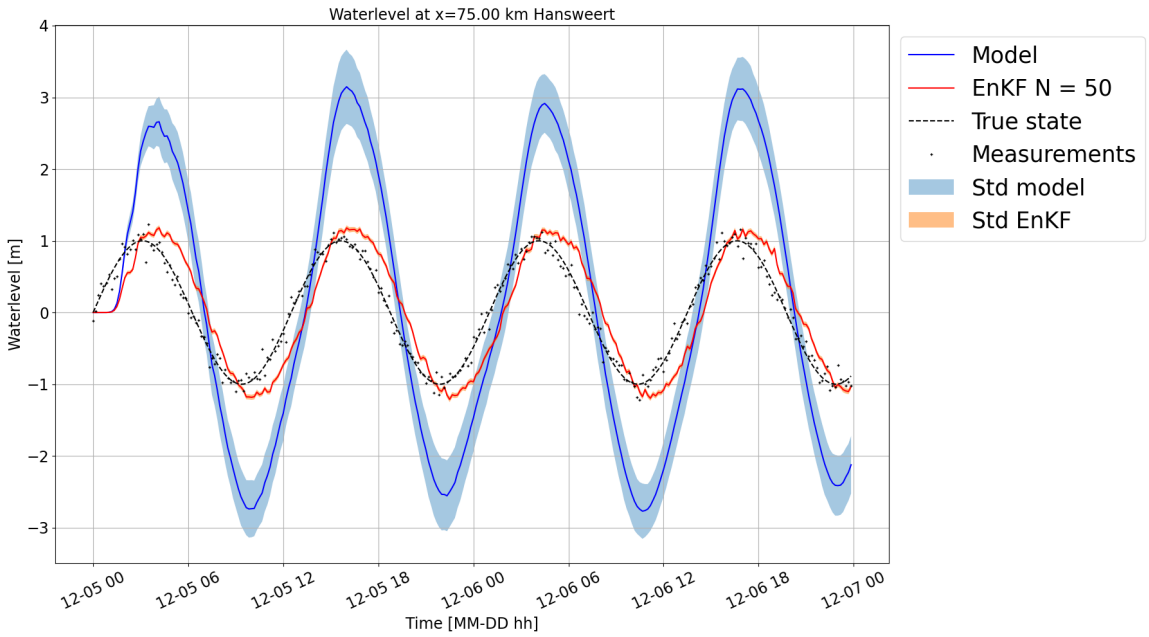


Figure 6.30: The waterlevel estimates at Hansweert over two days, computed with the hydrodynamic model and EnKF5 with generated measurements as in Equation 6.1 and furthermore with settings in Table 6.1 over two days from 5th of December until 7th of December. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model. Black dots: generated measurements.

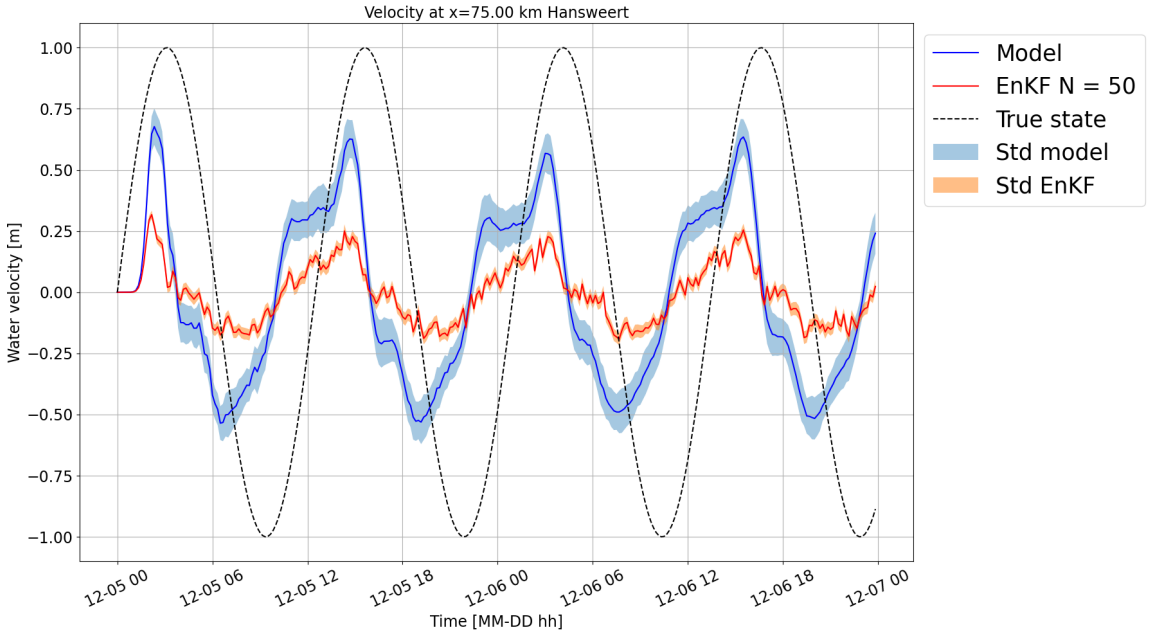


Figure 6.31: The velocity estimates at Hansweert over two days, computed with the hydrodynamic model and EnKF5 with generated measurements as in Equation 6.1 and furthermore with settings in Table 6.1 over two days from 5th of December until 7th of December. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model.

The rmse of all experiments with other generated observations are given in Figure 6.32. First of all, it is noticed that the rmse of the 'sin' experiment for velocity is very large. In that way, even worse than the model estimates. Also the rmse of delay is quite large compared to the model rsme. The rmse for 'offset' for velocity is also larger than the model rmse. It can be concluded that observations should be

close to the model. Therefore, models that already predict the true state to some extend are preferred, which is intuitive.

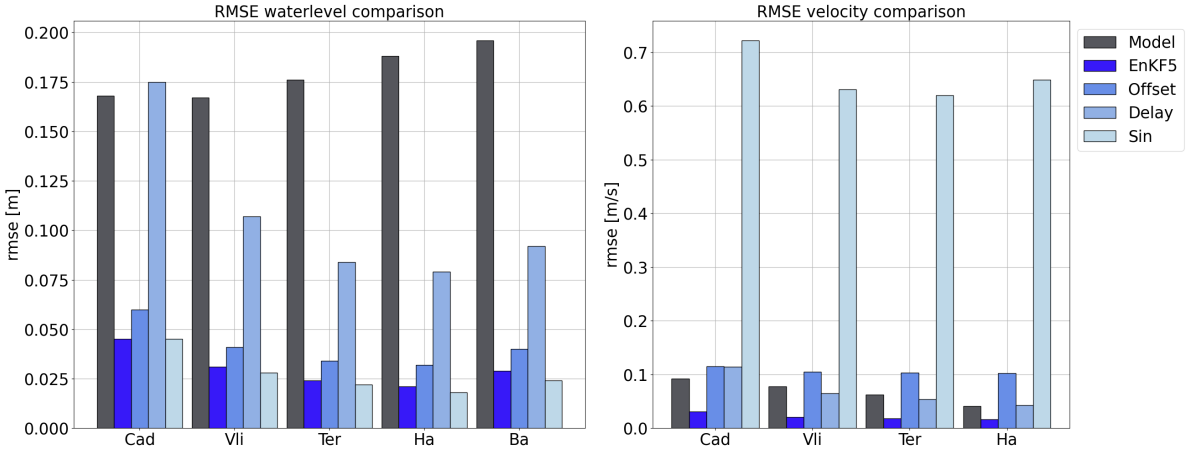


Figure 6.32: Left: rmse of the waterlevel h estimate per measuring station per model with generated observations for experiments EnKF5, Offset, Delay and Sin. Right: rmse of the velocity u prediction per measuring station per model with generated observations for experiments EnKF5, Offset, Delay and Sin.

7

2D model

In this chapter, the twin experiments mentioned in chapter 5 are performed with the Ensemble Kalman filter (section 4.3) on a 2D hydrodynamic model with depth made with Delft3D FM (section 3.5) for a simplified Western Scheldt estuary model. In this model, waterlevel, velocity and temperature are taken into account. The model and experiments in this chapter are performed to get a better understanding of the workings Delft3D FM and OpenDA and to understand the implementation of both. Furthermore, the effect of assimilation of water velocity and temperature on the water surface on the depth is investigated. Since usually observations are only present at the water surface, knowing if estimates within the water column improve by assimilation on the water surface is useful for better prediction of estimates within the water column. To that end, the detailed setup is mentioned in section 7.1, the performed experiments are discussed in section 7.2 and the results are shown in section 7.3.

7.1. Setup 2D model Western Scheldt estuary

A simplified grid for the Western Scheldt estuary is assumed, see Figure 7.1 and Figure 7.2. In this grid of approximately 100 km long and 0.5 km wide, 3 measuring stations are considered, called Station01, Station02 and Station03, at respectively 30 km, 60 km and 90 km. The grid has assumed depth of -8 m at the left boundary linearly increasing to -5 m at the right boundary below the average water surface at 0 m added with 20 σ-layers (subsection 3.5.1). The model used on this grid is implemented in Delft3D FM, with settings in Table 7.1. Wind is taken into account with 5 m/s in both x and y-direction, furthermore humidity is set on 70 %, the air temperature at 30 °C and cloudiness at 10 %. The boundary conditions are an astronomic boundary for waterlevel with 2 constituents is initialized on the eastboundary at 99 km. A discharge of 500 m^3/s is initialized on the left boundary at 0 km. It is assumed that waterlevel h, water velocity u and temperature T are measured at the measuring stations and thus that the state vector contains waterlevel h (s1 in Delft3D FM), water velocity u (unorm in Delft3D FM), and temperature T (tem1 in Delft3D FM). The true state and observations are generated as explained in section 5.2.

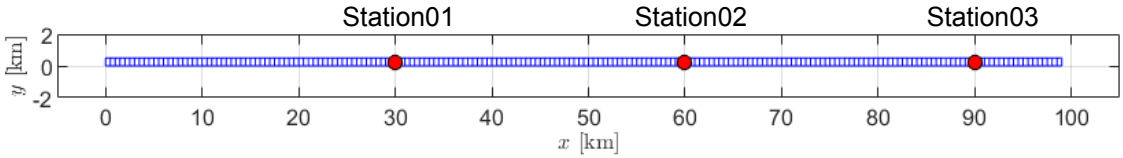


Figure 7.1: The grid of the 2D model, view of the xy-plane. There are 199 nodes in the x-direction and 2 in the y-direction, with a regular grid size (resolution) of 500 m. The red dots represent 3 measuring stations, called Station01, Station02, Station03, at respectively $(30000$ m$, 250$ m$)$, $(60000$ m$, 250$ m$)$ and $(90000$ m$, 250$ m$)$. The grid in the z-direction has 20 σ-layers, with layer 1 at the bedlevel and layer 20 at the water surface. The bed level varies from -8m at the left boundary to -5 m at the right boundary.

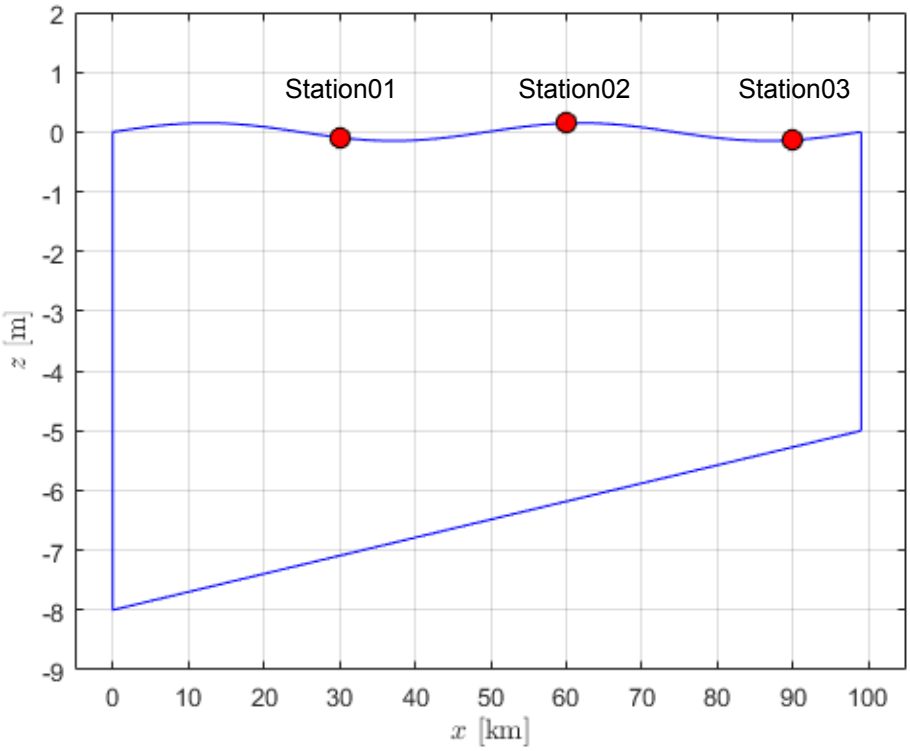


Figure 7.2: The grid of the 2D model, view of the xz-plane. There are 199 nodes in the x-direction and 2 in the y-direction, with a regular grid size (resolution) of 500 m. The red dots represent 3 measuring stations, called Station01, Station02, Station03, at respectively $(30000\text{ m}, 250\text{ m})$, $(60000\text{ m}, 250\text{ m})$ and $(90000\text{ m}, 250\text{ m})$. The grid in the z-direction has 20 σ-layers, with layer 1 at the bedlevel and layer 20 at the water surface. The bed level varies from -8 m at the left boundary to -5 m at the right boundary.

Within the OpenDA software environment, the settings from Delft3D FM are retrieved and the model can be run deterministic or stochastic. A run of the deterministic model will result in estimates of the waterlevel h, water velocity u and temperature T on the whole grid in Figure 7.1 and Figure 7.2 from 1991-01-01 00:00 to 2 days later, see Figure 7.3 for the surface estimates and Figure 7.4 for the bedlevel estimates. In Appendix D, temperature and water velocity estimates in intermediate σ-layers: 4, 8, 12 and 16 are shown, to see how temperature and water velocity estimates change in the water column. Waterlevel and water velocity have a periodic behavior in time caused by tidal propagation, while the temperature is increasing with increasing time.

Furthermore, deterministic velocity and temperature depth profiles from 1991-01-01 00:00 to 2 days later at Station01 are shown in Figure 7.5, to see the estimate at one location. In addition, deterministic velocity and temperature depth profiles from 1991-01-01 00:00 to 2 days later in Station02 and Station03 are shown in Appendix D. Within the water column at a specific time t, water velocity and temperature do not vary much. However, water velocity increases with increasing σ-layer. This is expected since water velocity is largest at the water surface [19]. Plotting on the whole grid is only possible for deterministic model runs, when there is no need for plotting error estimates. To that end, for stochastic model runs, estimates with error estimates at separate locations are considered.

The computational time of the software for the stochastic model is an issue. The time OpenDA needs to perform a model run depends on the number of ensemble members of the algorithm, the total time, time step and the computational time to perform a model run with the external software. In this thesis, it has been found that the computational time of running the model with settings in Table 7.1 doubles when doubling the number of ensembles, see Table 7.2. It is chosen to produce model runs with OpenDA with number of ensemble members of $N = 16$.

Table 7.1: Settings of the 2D model in the MDU file. All other parameters are set to their default values, found in [28].

Setting	Value	Unit
WaterLevIni	1	m
Bedlevuni	-5	m
Bedslope	0	-
AngLat	52	-
AngLon	0	-
Kmx	20	-
Layertype	1	-
Vicouv	10	$\frac{m^2}{s}$
Dicouv	10	$\frac{m^2}{s}$
TidalForcing	1	-
Temperature	5	-
InitialTemperature	6	°C
Secchidepth	1	m
Stanton	1.3d-3	-
Dalton	1.3d-3	-
RefDate	19910101	-
Tunit	M	-
DtUser	60	s
TStart	0	Tunit
TStop	2880	Tunit
WindExt	1	-
HisInterval	3600	s
MapInterval	3600	s
Wrihis_temperature	1	-
Wrimap_waterlevel_s0	1	-
Wrimap_waterlevel_s1	1	-
Wrimap_velocity_component_u0	1	-
Wrimap_velocity_component_u1	1	-
Wrimap_velocity_vector	1	-
Wrimap_flow_flux_q1	1	-
Wrimap_heat_fluxes	1	-
Wrimap_temperature	1	-

Table 7.2: Computational time of openDA for the 2D model, depending on the number of ensemble members.

Number of ensemble members	Computational time
$N = 4$	3 min
$N = 8$	7 min
$N = 16$	15 min
$N = 32$	30 min
$N = 64$	1 hour
$N = 128$	2 hours

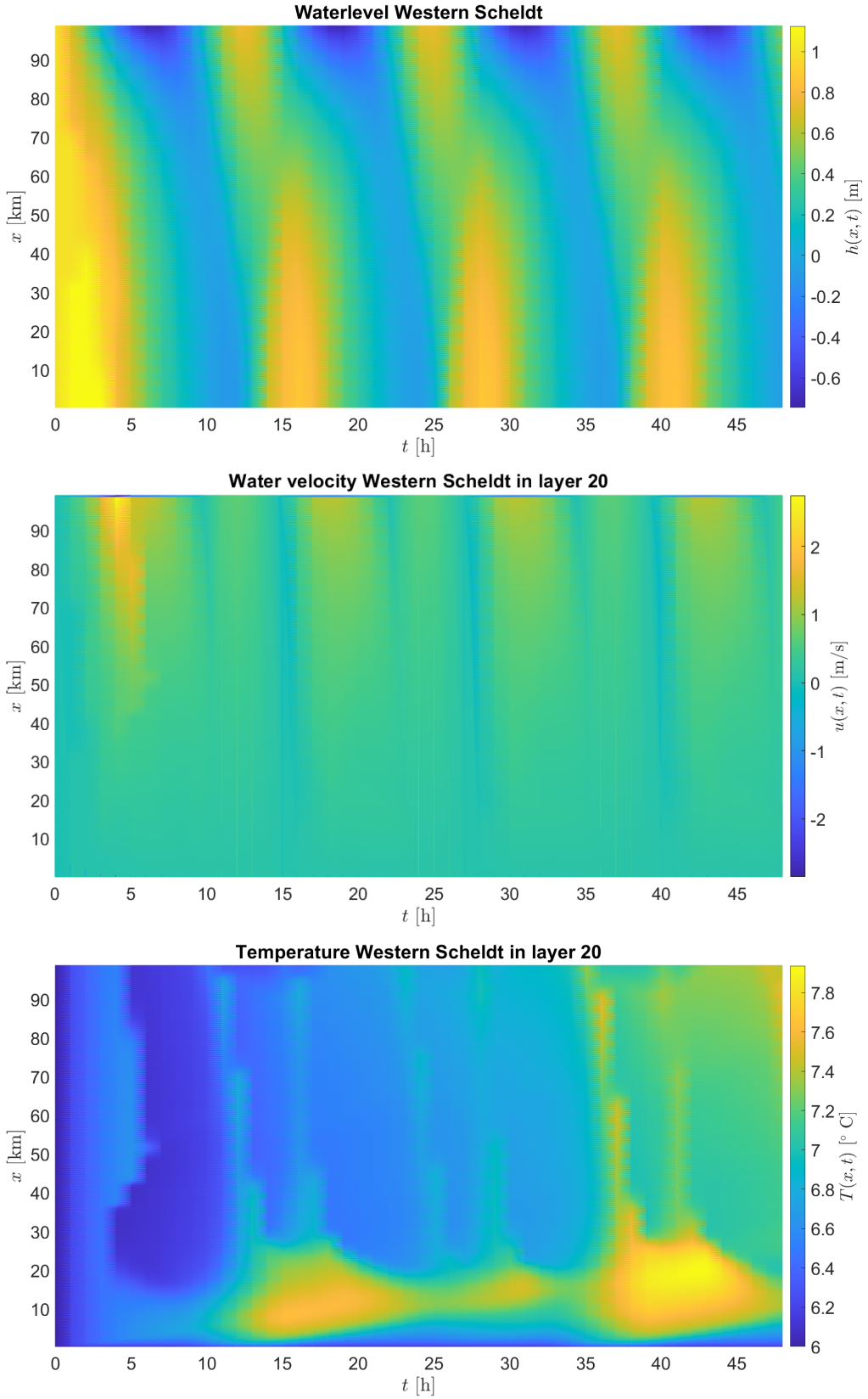


Figure 7.3: Deterministic estimates of the 2D estuary model, plotted as a function of position x and time t after 1991-01-01 00:00. Top: Waterlevel $h(x,t)$. Middle: Velocity $u(x,t)$ in the surface with σ-layer 20. Bottom: Temperature $T(x,t)$ in the surface layer with σ-layer 20.

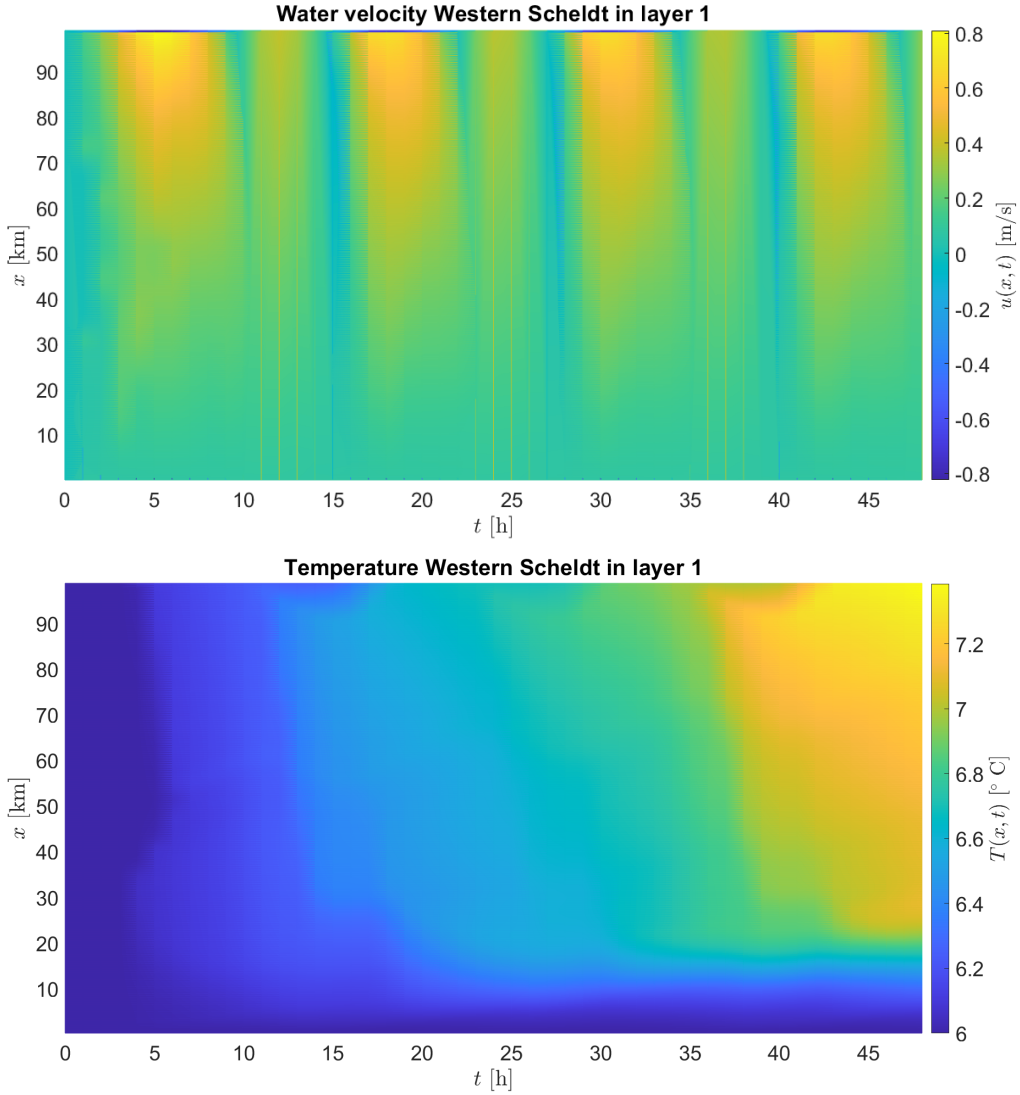


Figure 7.4: Deterministic estimates of the 2D estuary model, plotted as a function of position x and time t after 1991-01-01 00:00. Top: Velocity $u(x,t)$ at the bedlevel with σ-layer 1. Bottom: Temperature $T(x,t)$ at the bedlevel with σ-layer 1.

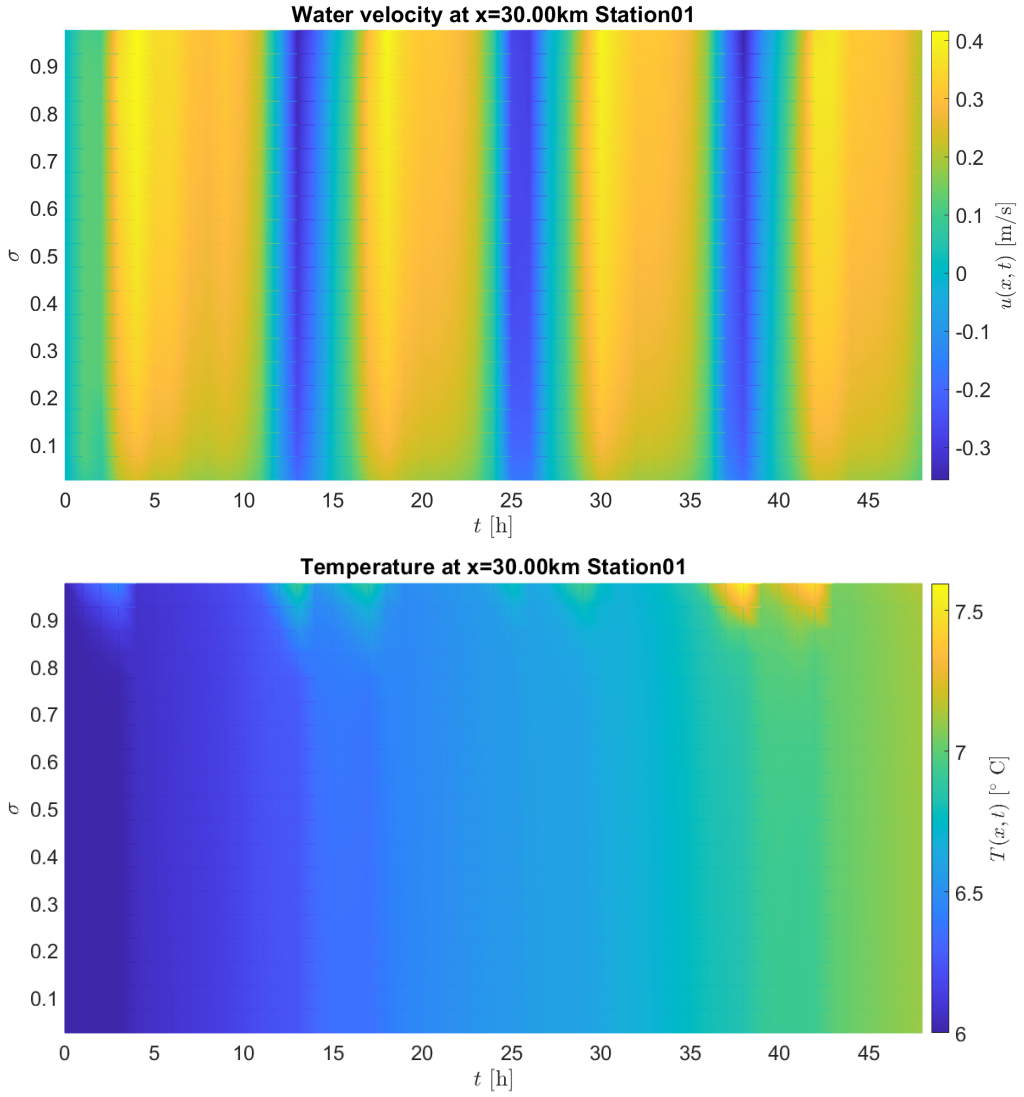


Figure 7.5: Deterministic estimate of the depth profile of velocity $u(x,t)$ and temperature $T(x,t)$ at Station01 over two days after 1991-01-01 00:00, plotted as a function of time t and σ-layers, where 1 is the water surface and 0 the bed level divided in 20 layers.

For stochastic model runs, noise is set on the east boundary condition for waterlevel of standard deviation 0.2. A run of the stochastic 2D model with $N = 16$ ensembles at Station01 is depicted in Figure 7.6 top for waterlevel, middle for water velocity and bottom for water temperature.

As can be seen by comparing the figures in Figure 7.6, the standard deviation on the waterlevel, water velocity and temperature is present. However, the standard deviation on the temperature is not really constant over time. This is the case since no noise on the temperature is set, while only noise on the boundary of the waterlevel is set and this propagates into the water velocity and temperature. In this thesis, the effect of assimilation of surface parameters on the estimate within the water column is sought for. To be able to assimilate observations into the model, it is required that the standard deviation is more constant. Additionally, [61] advocates the importance of having constant standard deviation for temperature assimilation experiments, as the EnKF may collapse due to very low standard deviation on EnKF. Sequential data assimilation algorithms require a stochastic version of the estimate. Spatially correlated noise may be set on the temperature flux, in OpenDA specified as heat flux, to tackle this. Due to time constraints for this thesis, it has not been accomplished to set noise on the temperature. Therefore, assimilation of temperature at the water surface is investigated, but the results may be inaccurate. Additionally, assimilation of water velocity at the water surface is investigated.

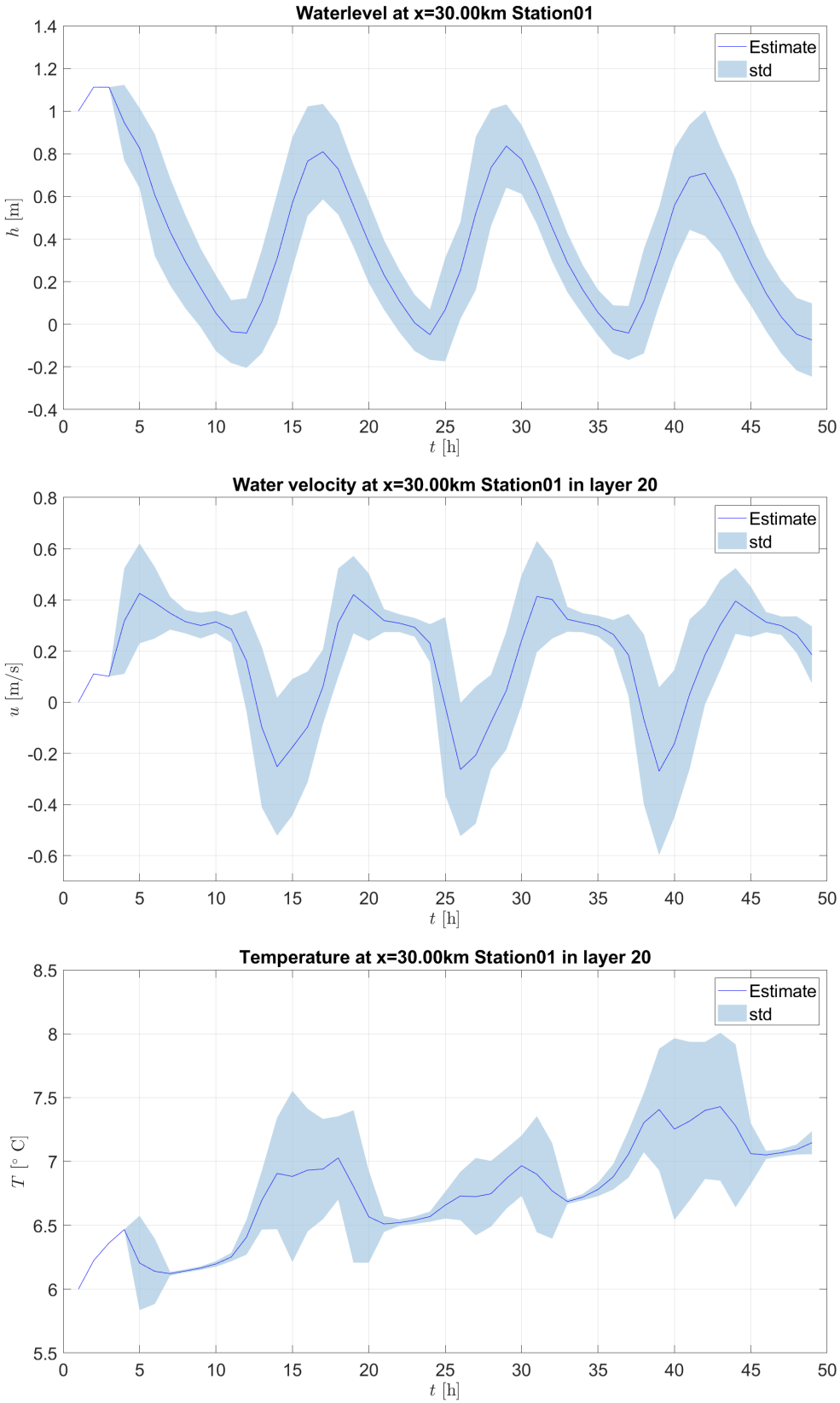


Figure 7.6: Stochastic estimates of the 2D estuary model over two days starting at 1991-01-01 00:00. Top: Waterlevel h. Middle: Water velocity u at the water surface with σ-layer 20. Bottom: Temperature T at the water surface with σ-layer 20. Computed with Delft3D FM and OpenDA.

7.2. Experiment setup

The twin experiments to investigate the assimilation of observations with the Ensemble Kalman filter on the model specified in section 7.1 are mentioned. Especially, the focus lies on assimilation of water surface parameters, and the effect on the estimates within the water column. The experiments are given here:

1. **Effect of assimilation location**
   This experiment, related to experiment 3 in section 6.2, will test the effect of assimilation at different locations. To that end, waterlevel observations are assimilated at Station01, Station02, Station03 and all possible combinations of these stations, to see if the filter improves the estimate. The true state is chosen from the stochastic model and observations are generated as explained in section 5.2 with $\mathbf{R}_k = 0.01\mathbf{I}$.

2. **Effect of observation error covariance $\mathbf{R}_k$**
   This experiment, related to experiment 5 in section 6.2, will test the effect of the observation error covariance $\mathbf{R}_k$ on the estimate. The model error covariance is kept at a value of 0.2, while the observation error covariance $\mathbf{R}_k$ is varied from $0.01\mathbf{I}$ to $\mathbf{I}$. Both values of observation error covariance below and above the model error covariance are chosen. This is done to analyze the effect of the Kalman Gain between the model error covariance and observation error covariance on the estimate. The true state is chosen from the stochastic model and observations are generated as explained in section 5.2 with varying $\mathbf{R}_k$.

3. **Effect of surface assimilation of water velocity on estimate within water column**
   Since the 2D model has depth included, experiments are setup to investigate if the Ensemble Kalman filter can improve the estimation of physical parameters within the water column. This is only possible for physical variables other than waterlevel. Water velocity observations are assimilated on the surface at σ-layer 20 in Station01, Station02 and Station03. It is investigated if water velocity assimilation on the surface will improve the estimates in the water column. To that end, the estimates in the σ-layers: 4, 8, 12, 16 (within water column) and 20 (at the water surface) below the stations are investigated, see Figure 7.7. The true state is chosen from the stochastic model and observations are generated as explained in section 5.2 with $\mathbf{R}_k = 0.01\mathbf{I}$.

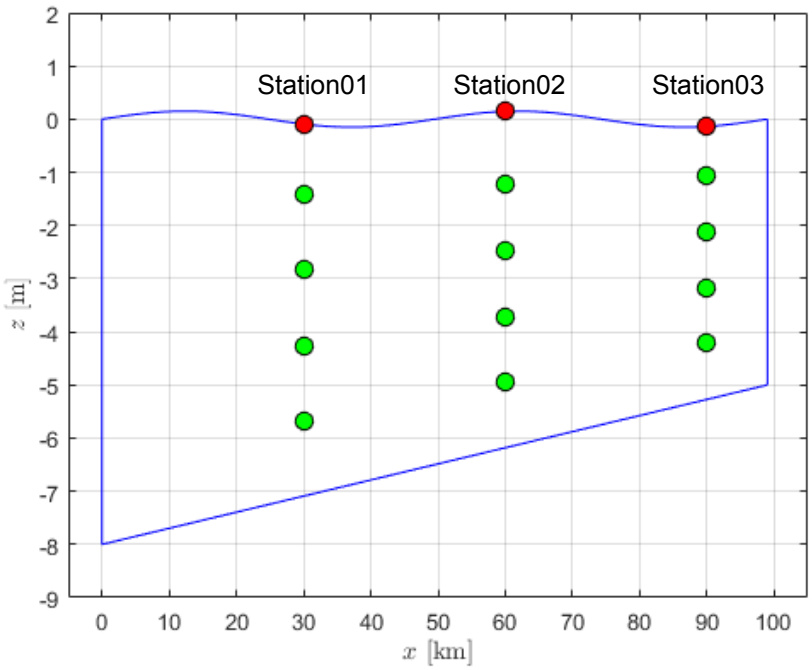


Figure 7.7: The grid of the 2D model, view of the xz-plane. Water velocity or temperature at the water surface (σ-layer 20) in Station01, Station02 and Station03 is assimilated (in red). The estimates of water velocity or temperature in σ-layers 4, 8, 12, 16 and 20 (in green and red) are investigated.

4. **Effect of surface assimilation of temperature on estimate within water column**
   Similarly to experiment 3, it is investigated if temperature assimilation on the surface will improve the estimates in the water column. Therefore, temperature observations are assimilated on the surface at σ-layer 20 in Station01, Station02 and Station03. The estimates in the σ-layers: 4, 8, 12, 16 (within the water column) and 20 (at the water surface) below the stations are investigated, see Figure 7.7. The true state is chosen from the stochastic model estimations and observations are generated as explained in section 5.2 with $\mathbf{R}_k = 0.01\mathbf{I}$.

7.3. Results

In this section, the three experiments mentioned in section 7.2 are executed and the results shown and examined.

In Figure 7.8, the model prediction and chosen true state for waterlevel for Station01 is shown. Indeed, the model prediction has standard deviation of around 0.2 m. The true state is clearly taken from the model due to the prediction laying in the standard deviation zone from the model estimate.

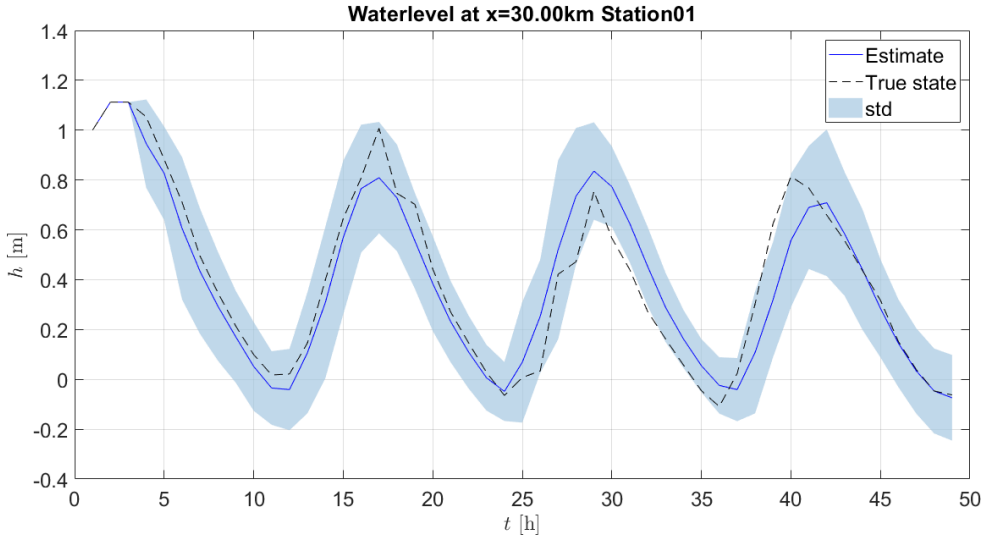


Figure 7.8: The waterlevel estimates over two days starting at 1991-01-01 00:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Black striped: chosen true state from the model.

Again, many estimates of performed EnKF model runs could be shown showing the estimates of waterlevel, water velocity and temperature, where only a few will be shown here. In experiment 1 and 2, only waterlevel is considered. The EnKF estimations for assimilation of waterlevel observations at Station01, Station02 and Station03 with $\mathbf{R}_k = 0.01\mathbf{I}$ are shown in Figure 7.9. The figures show that the estimate of waterlevel in Station01 and Station02 are very accurate. However, the true state and thus the EnKF estimate do not seem smooth, due to the many non-differentiable function pieces. It can be argued that the true state is not generated smoothly in OpenDA. For Station03, the standard deviation of the EnKF prediction is larger than in Station01 and Station02. Although the EnKF clearly gives a better estimation, the error in Station03 is higher.

In Figure 7.10, the EnKF estimations for assimilation of waterlevel observations at Station01, Station02 and Station03 with $\mathbf{R}_k = 0.5\mathbf{I}$ are shown. Since the observations are relatively far away from the true state, it is expected that the predictions of the EnKF will not be much better than the model predictions. This is also seen in Figure 7.10, although it seems the EnKF does change in estimation from the model.

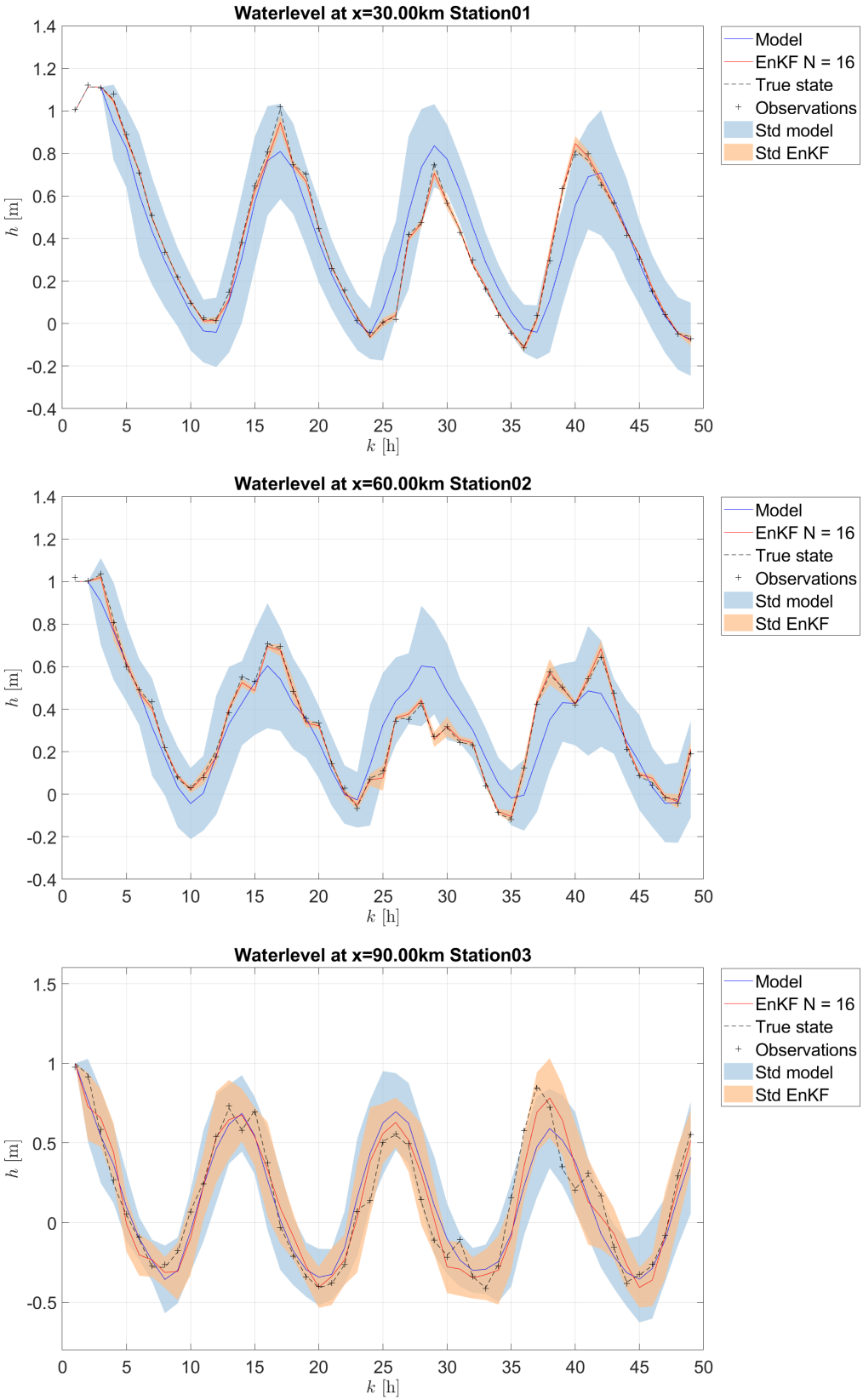


Figure 7.9: The waterlevel estimates over two days starting at 1991-01-01 00:00. Top: Station01. Middle: Station02. Bottom: Station03. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF with assimilated waterlevel observations at Station01, Station02 and Station03 with $\mathbf{R}_k = 0.01\mathbf{I}$. Yellow: standard deviation of the EnKF estimate. Black striped: chosen true state from the model. Black crosses: generated observations.

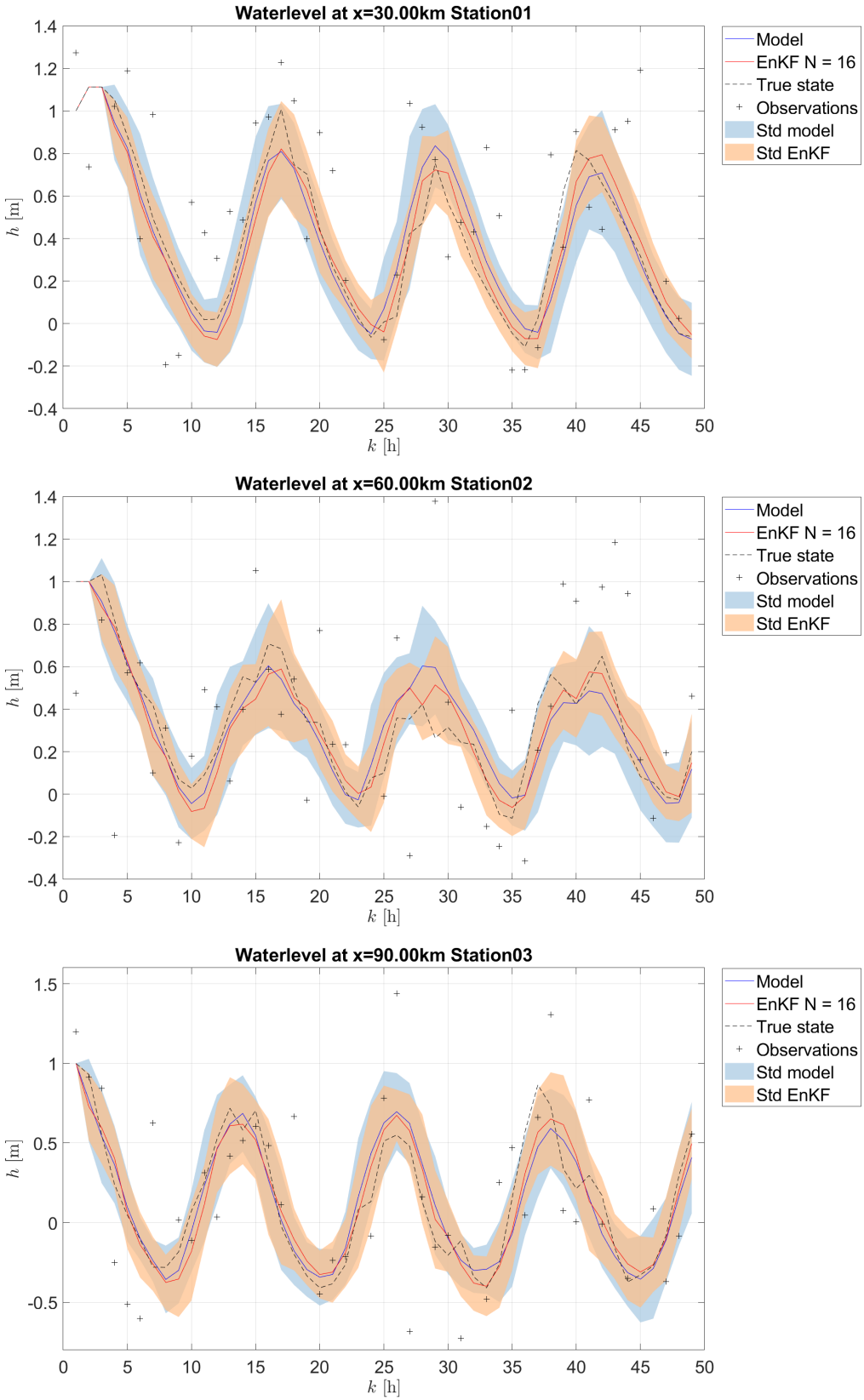


Figure 7.10: The waterlevel estimates over two days starting at 1991-01-01 00:00. Top: Station01. Middle: Station02. Bottom: Station03. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF with assimilated waterlevel observations at Station01, Station02 and Station03 with $\mathbf{R}_k = 0.5\mathbf{I}$. Yellow: standard deviation of the EnKF estimate. Black striped: chosen true state from the model. Black crosses: generated observations.

7.3.1. Effect of assimilation location

This experiment, related to experiment 3 in section 6.2, will test the effect of assimilation at different locations. To that end, waterlevel observations are assimilated at Station01, Station02, Station03 and all possible combinations of these stations, to see if the filter improves the estimate. The true state is chosen from the stochastic model and observations are generated as explained in section 5.2 with $\mathbf{R}_k = 0.01\mathbf{I}$. The rmse between the true state and the model and EnKF predictions are shown in Figure 7.11.

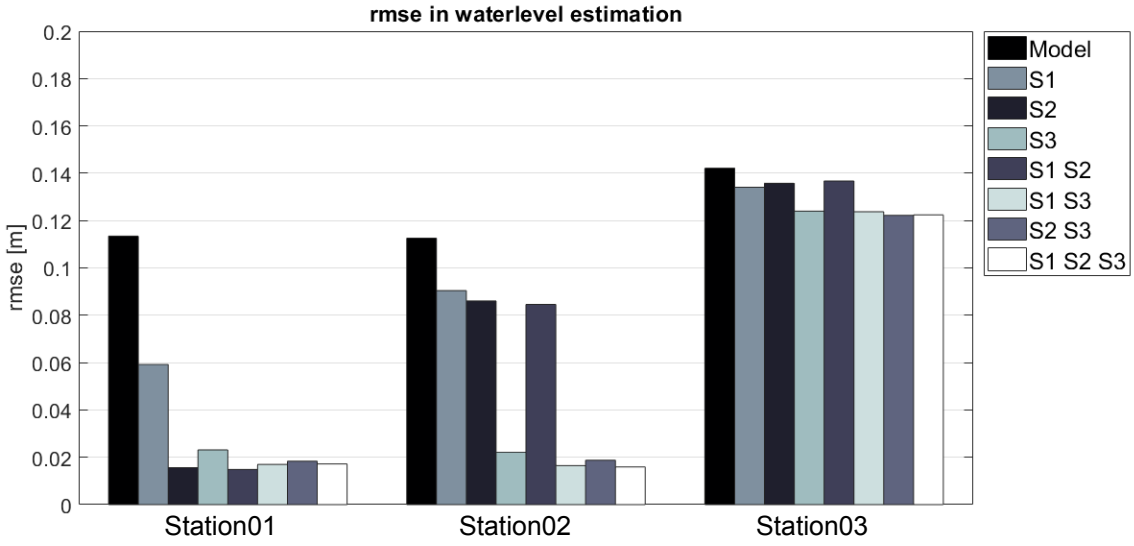


Figure 7.11: rmse in waterlevel h estimation per measuring station for the model and EnKF for assimilation of waterlevel h at respectively S1 (Station01), S2 (Station02), S3 (Station03), S1 S2, S1 S3, S2 S3 and S1 S2 S3 with $\mathbf{R}_k = 0.01\mathbf{I}$.

All EnKF estimates have smaller rmse than the model, and thus all EnKF estimates give a more accurate estimate than the model. Furthermore, at Station01 and Station02 the assimilation of observations at multiple station gives a better estimate. Most important for the estimation at Station02 is the assimilation of observations at Station03, which gives the greatest reduce in rmse. For the estimation at Station01, both assimilation of observations at Station02 and Station03 is most beneficial, while assimilation of only observations at Station01 reduces the rmse by a half. The rmse in estimation at Station03 does not reduce as much as in Station01 or Station02. The hypothesis it that this the case due to the fact how the model propagates the noise through the water in the x-direction. As the noise is set on the east boundary near to Station03, more noise is coming into the water column than the EnKF can keep up with.

7.3.2. Effect of observation error covariance

This experiment, related to experiment 5 in section 6.2, will test the effect of the observation error covariance $\mathbf{R}_k$ on the estimate. The model error covariance is kept at a value of 0.2, while the observation error covariance $\mathbf{R}_k$ is varied from $0.01\mathbf{I}$ to $\mathbf{I}$. Both values of observation error covariance below and above the model error covariance are chosen. This to see the effect of the Kalman Gain between the model error covariance and observation error covariance on the estimate. The true state is chosen from the stochastic model and observations are generated as explained in section 5.2 with varying $\mathbf{R}_k$. The rmse between the true state and the model and EnKF predictions are shown in Figure 7.12.

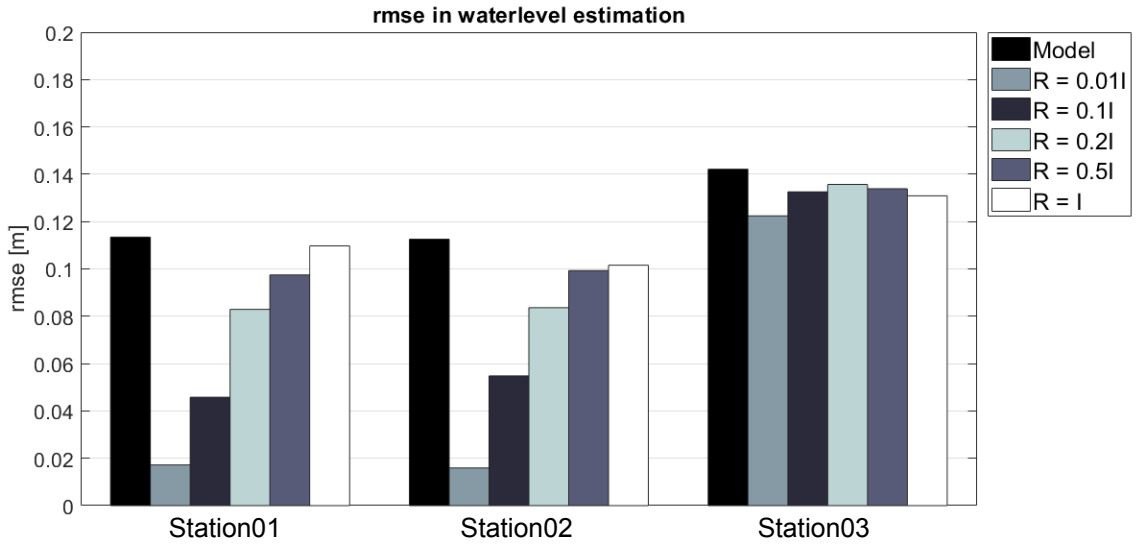


Figure 7.12: rmse in waterlevel h estimation per measuring station for the model and EnKF for assimilation of waterlevel h at Station01, Station02 and Station03 with varying $\mathbf{R}_k$.

All EnKF estimates have smaller rmse than the model, and thus all EnKF estimates give a more accurate prediction than the model. Furthermore, at Station01 and Station02 the assimilation of observations with decreasing $\mathbf{R}_k$ gives an decreasing rmse and thus a better estimation. Again as discussed in subsection 7.3.1 the estimation at Station03 does not improve much by assimilation of observations. For Station01 and Station02, assimilation has a big effect on the accuracy.

7.3.3. Effect of surface assimilation of u on estimate within water column

Water velocity observations are assimilated at the water surface at Station01, Station02 and Station03. Time series of water velocity model estimates and water velocity estimates with EnKF at Station01 and Station03 for σ-layers 20 (at water surface), 12 and 4 (within water column) are shown in Figure 7.13 and Figure 7.14 respectively. When comparing both Figure 7.13 and Figure 7.14, it stands out that the EnKF estimate has a smaller standard deviation for Station01 than Station03, as seen as well in experiment 1 and 2 for the estimate in waterlevel. Where the EnKF gets the estimate close to the true state at Station01 (and similarly for Station02), the estimate at Station03 is a bit less accurate. Furthermore, assimilation of water velocity observations at the water surface seems to predict the conditions in the water column quite well for Station01, as all EnKF estimates seem to lay closer to the true state.

In Figure 7.15, the rmse in water velocity model and EnKF estimate for different σ-layers is displayed. The EnKF estimate has always lower rmse than the model estimate counterpart. For Station01 and Station02, the rmse for estimate with EnKF is two times lower than the model estimate counter part. For Station03, the rmse for EnKF is only slightly lower as discussed earlier. To conclude, it may be beneficial to assimilate observations at the water surface to get a two times more accurate estimate within the water column.

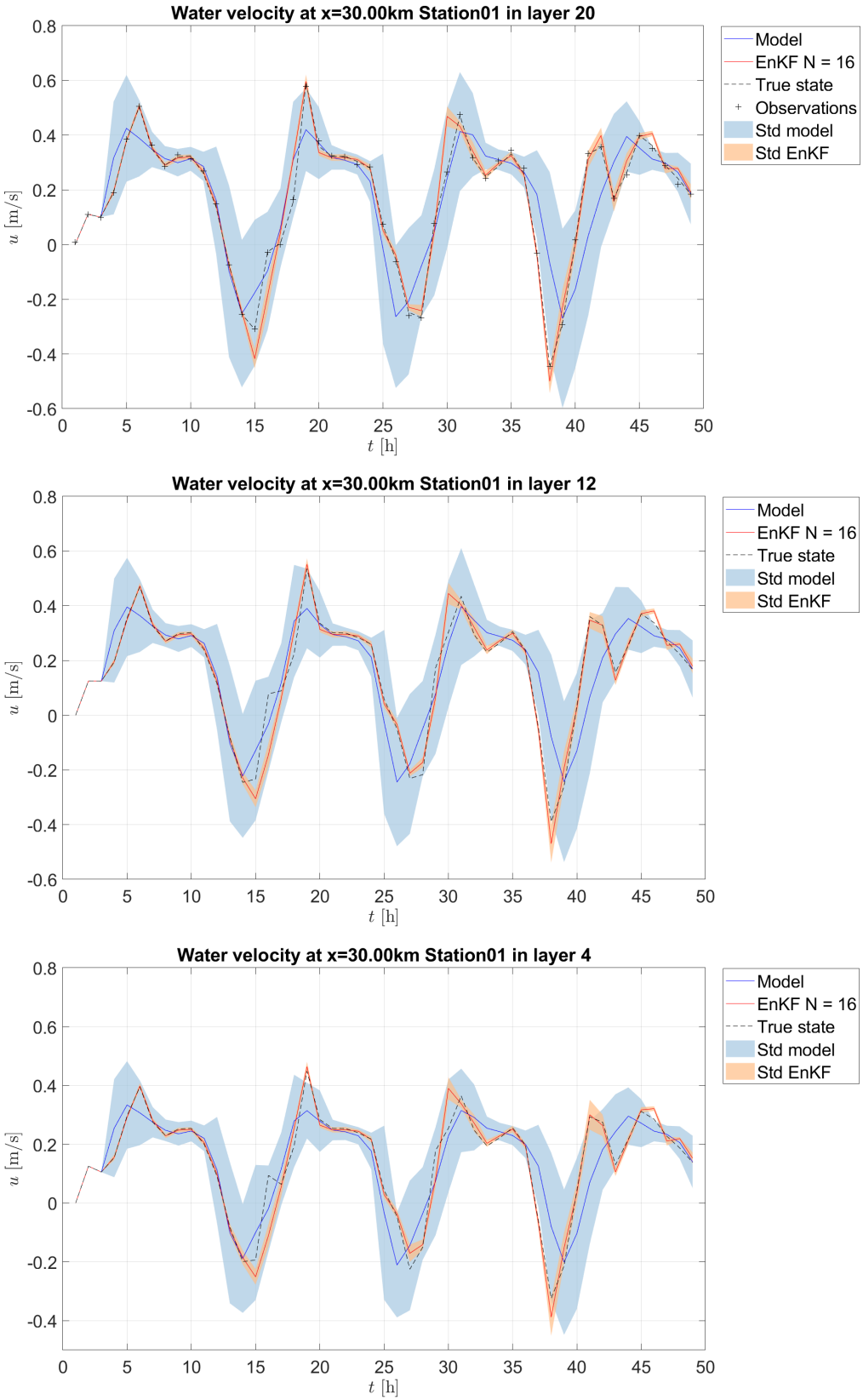


Figure 7.13: The water velocity estimates at Station01 over two days starting at 1991-01-01 00:00. Top: σ-layer 20 (water surface). Middle: σ-layer 12. Bottom: σ-layer 4. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF with assimilated water velocity observations at σ-layer 20 in Station01, Station02 and Station03. Yellow: standard deviation of the EnKF estimate. Black striped: chosen true state from the model. Black crosses: generated observations.

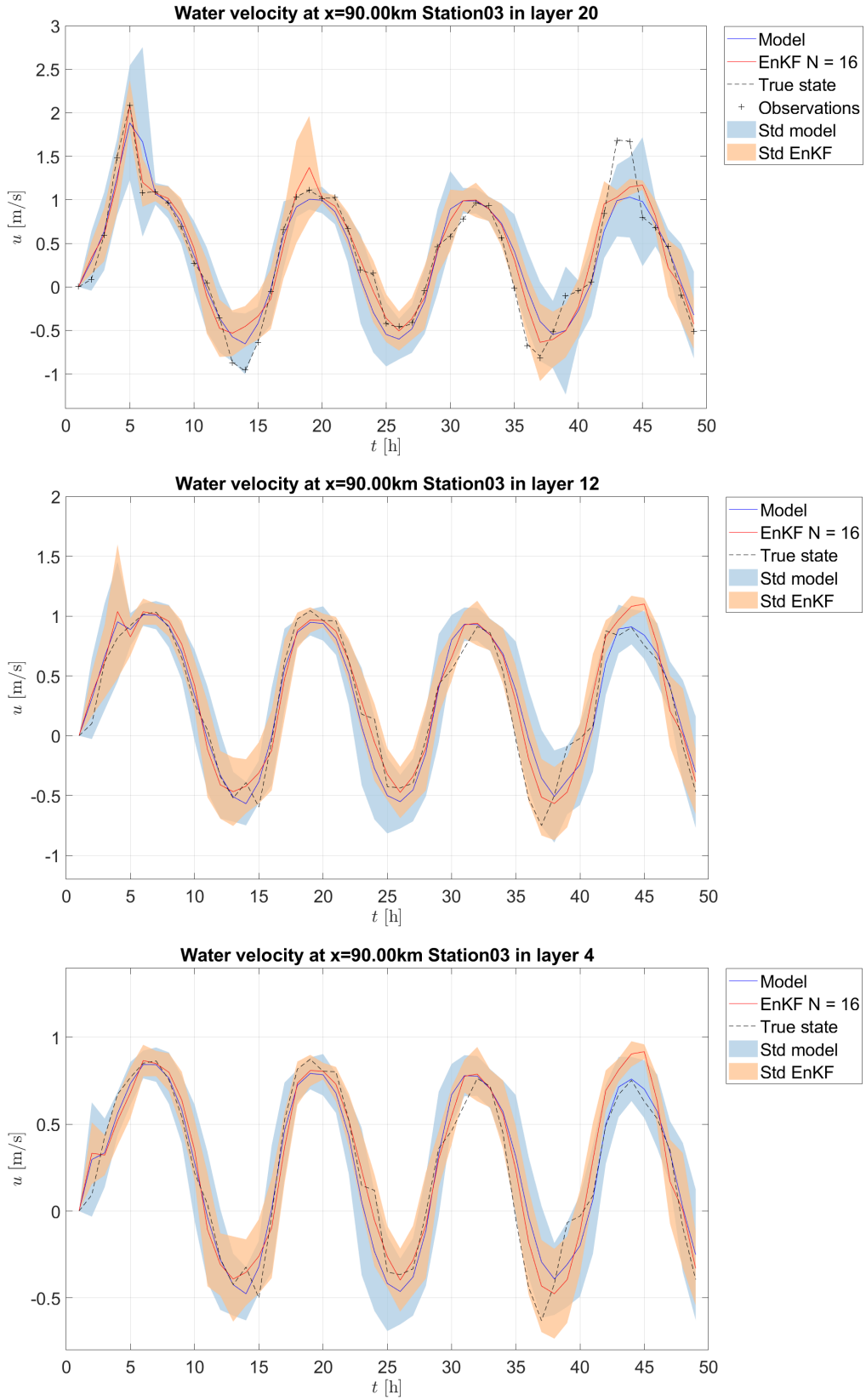


Figure 7.14: The water velocity estimates at Station03 over two days starting at 1991-01-01 00:00. Top: σ-layer 20 (water surface). Middle: σ-layer 12. Bottom: σ-layer 4. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF with assimilated water velocity observations at σ-layer 20 in Station01, Station02 and Station03. Yellow: standard deviation of the EnKF estimate. Black striped: chosen true state from the model. Black crosses: generated observations.

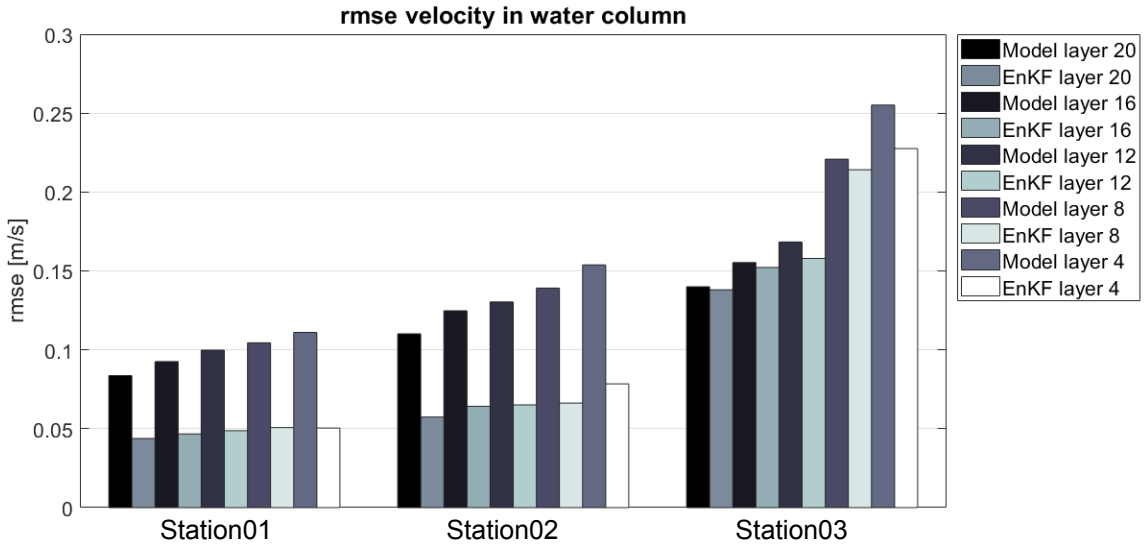


Figure 7.15: rmse in water velocity u estimate for the model and EnKF for assimilation of water velocity u at Station01, Station02 and Station03 in σ-layer 20. The rmse for both model and EnKF is shown at different σ-layers: 4, 8, 12, 16, 20, per measuring station Station01, Station02 and Station03.

As already stated in subsection 6.3.3, assimilation of velocity u does not result in a significant better estimate of the waterlevel h. Figure 7.16 shows the rmse per measuring station in waterlevel estimate for the model without EnKF and EnKF with water velocity at the water surface assimilated. The values of the rmse in Figure 7.16 comply with this statement, the estimates become slightly better or slightly worse.

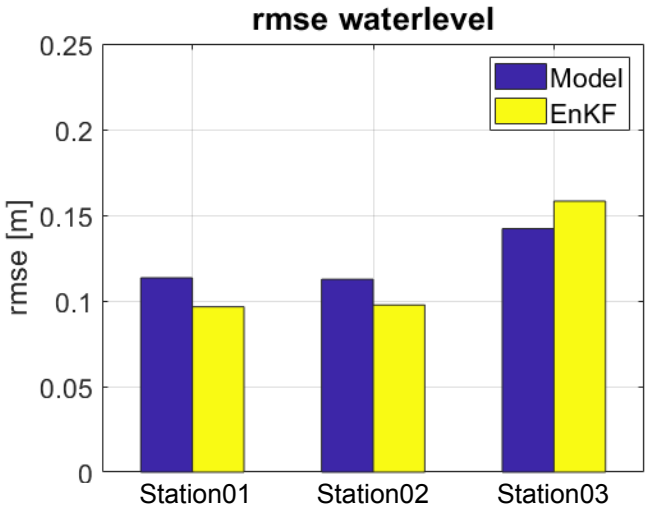


Figure 7.16: rmse in waterlevel h estimate for the model and EnKF for assimilation of water velocity u at Station01, Station02 and Station03 in σ-layer 20. Plotted per measuring station Station01, Station02 and Station03.

7.3.4. Effect of surface assimilation of T on estimate within water column

Similarly to experiment 3, time series of temperature model estimates and temperature estimates with EnKF estimates with assimilation of temperature at the water surface at Station01 and Station03 for σ-layers 20 (at water surface), 12 and 4 (within water column) are shown in Figure 7.17 and Figure 7.18 respectively.

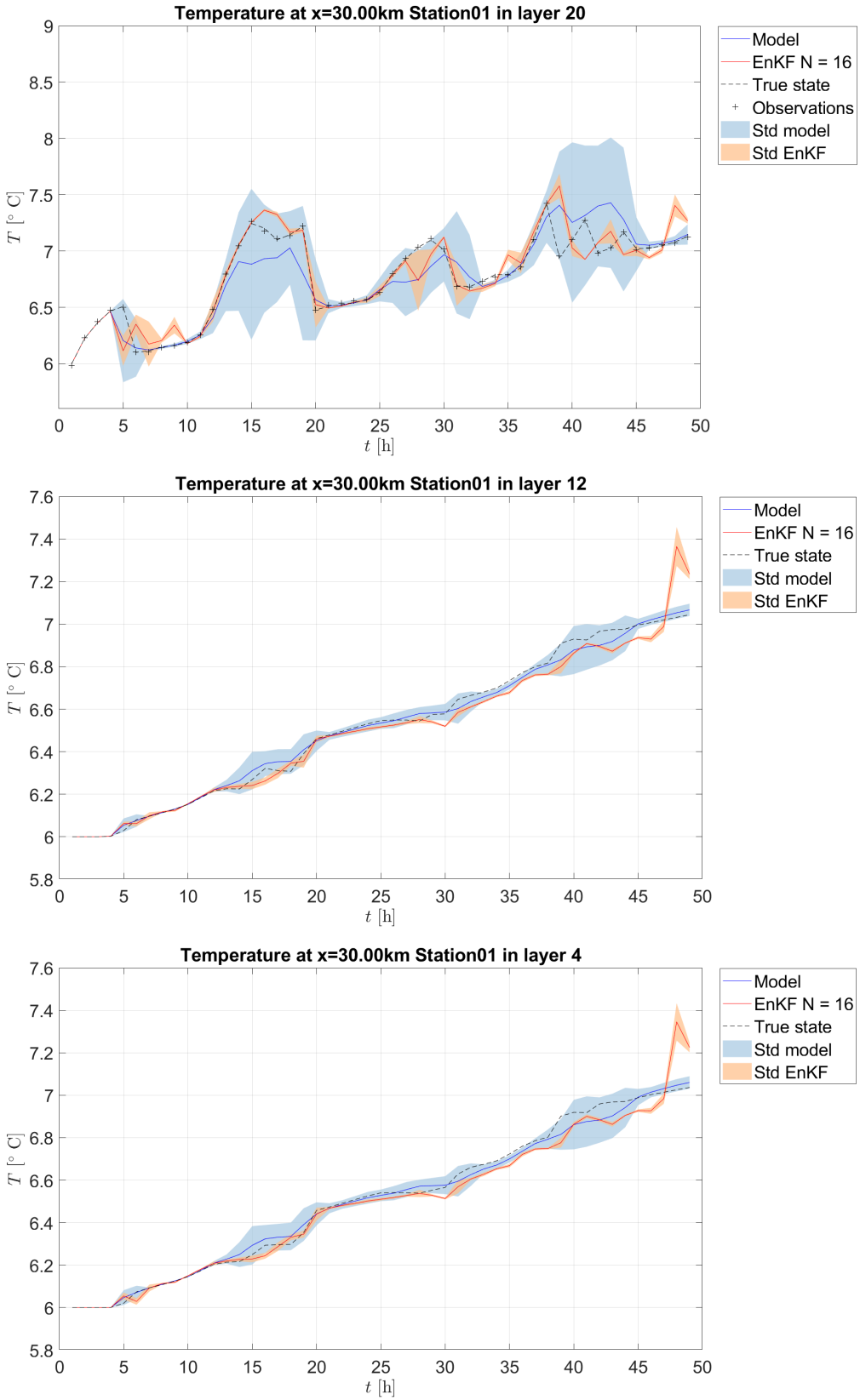


Figure 7.17: The temperature estimate at Station01 over two days starting at 1991-01-01 00:00. Top: σ-layer 20 (water surface). Middle: σ-layer 12. Bottom: σ-layer 4. Blue: The estimate of the model. Light blue: standard deviation of the model esimate. Red: Estimate of EnKF with assimilated temperature observations at σ-layer 20 in Station01, Station02 and Station03. Yellow: standard deviation of the EnKF estimate. Black striped: chosen true state from the model. Black crosses: generated observations.

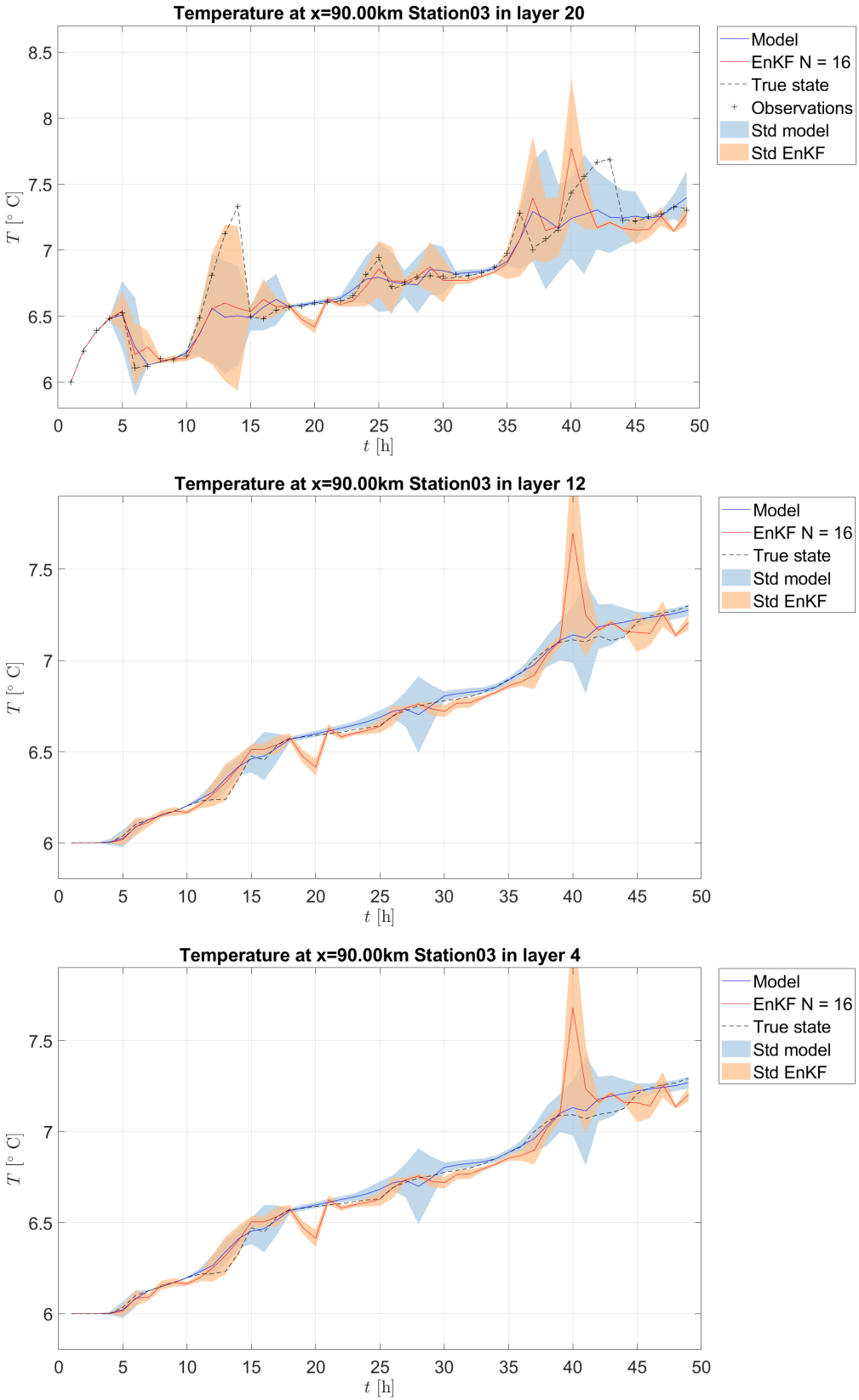


Figure 7.18: The temperature estimate at Station03 over two days starting at 1991-01-01 00:00. Top: σ-layer 20 (water surface). Middle: σ-layer 12. Bottom: σ-layer 4. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF with assimilated temperature observations at σ-layer 20 in Station01, Station02 and Station03. Yellow: standard deviation of the EnKF estimate. Black striped: chosen true state from the model. Black crosses: generated observations.

When comparing both Figure 7.17 and Figure 7.18, it stands out that the EnKF estimate has smaller standard deviation for Station01 than Station03, as seen as well in experiment 1 and 2 for the estimate in waterlevel. However, for both Figure 7.17 and Figure 7.18 at all σ-layers, the EnKF does deviate a lot from the true state. It seems that the EnKF cannot give a better estimate in this case. Only for σ-layer 20, where the observations are assimilated, the estimate is a bit closer to the true state than in other σ-layers.
In Figure 7.19, the rmse between the true state and model and EnKF estimates are shown. It confirms that the EnKF produces estimates with higher rmse than the model does. It seems for Station01, the rmse is two times as large, for Station02, only slightly larger, and for Station03 more than two times as large, at the same σ-layer.

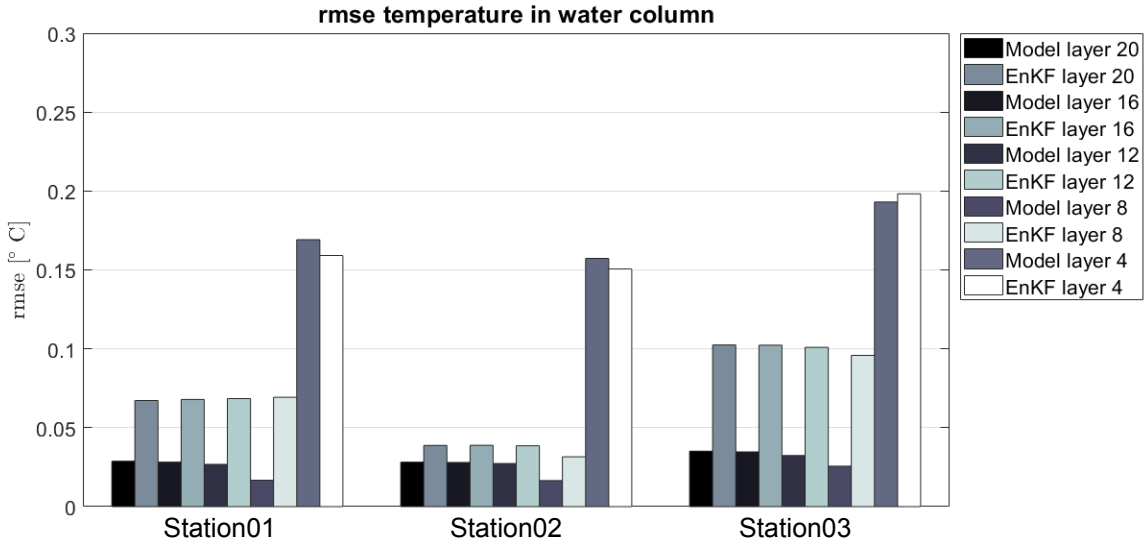


Figure 7.19: rmse in temperature T estimate for the model and EnKF for assimilation of temperature T at Station01, Station02 and Station03 in σ-layer 20. The rmse for both model and EnKF is shown at different σ-layers: 4, 8, 12, 16, 20, per measuring station Station01, Station02 and Station03.

These results may be due to the fact that the model standard deviation is not constant over the whole time period. The amount by which the model is updated is determined by the Kalman Gain (see Equation 4.5), or the trade-off between the model error covariance and the observational error covariance. When a sharp decrease in model error covariance occurs, the Kalman Gain quickly changes, which could interrupt clear estimate. This happens on occasion in the top figures in Figure 7.17 and Figure 7.18. However the most likely explanation is given by [61]. [61] states that the EnKF may collapse when a too-low standard deviation on the EnKF estimate is present. Indeed in every figure in Figure 7.17 and Figure 7.18, the EnKF estimates have very low standard deviation at least around time steps 10, 22 and 34. A hypothesis is that the EnKF thus collapses and therefore does not give good estimates.
[61] overcomes the collapsing filter by stopping the algorithm before it collapses, initiating a new ensemble from that time step with the results of the ended run and executing the model on this new ensemble. This may be researched in a later stage. However, it is advised in that case to set spatially correlated noise on the temperature to create noise with more constant standard deviation. The hypothesis is that this experiment may produce better estimates for temperature, as it does for water velocity.
Furthermore, the model itself may be tweaked further. The model has initial waterlevel of 6 °C, while the air temperature is 30 °C. This is not very realistic. More than two days may be examined as well, since the spin-up time for water temperature is larger than two days.

8

DCSM model

The 3D DCSM model is a hydrodynamic and water quality model, which covers the northwest European continental shelf, see Figure 8.1. There are different versions of this model. The one described here is the version made with an unstructured grid in Delft3D FM, see section 3.5.

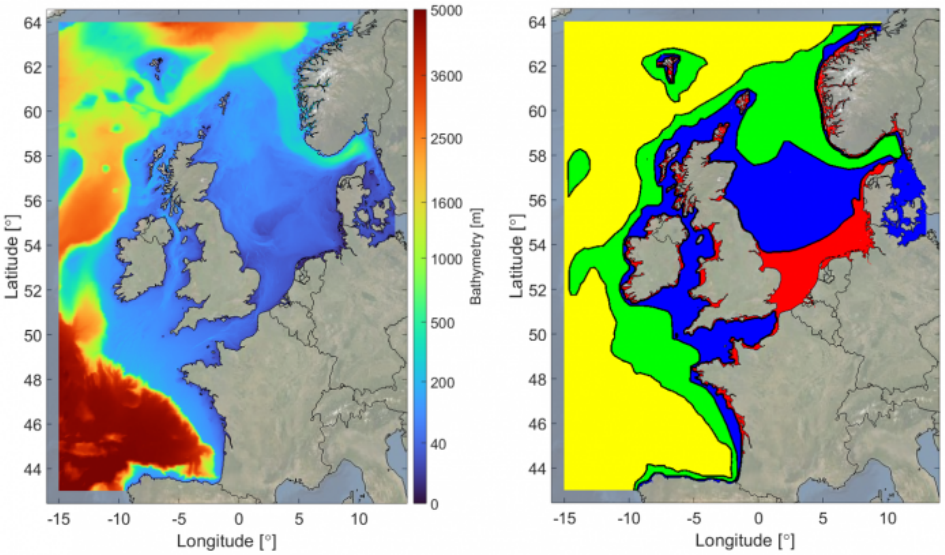


Figure 8.1: Left: Bathymetry (depth profile) of the DCSM model. Right: Resolutions (grid sizes) of the DCSM model. Bottom: Description colors of different resolutions within right figure. Figure retrieved from [27].

A detailed description of all settings in the model is given in [85, 88]. A few are mentioned here. The grid of the model consists of approximately 630,000 cells and is designed to have a resolution that increases with decreasing water depth. The differences in resolution allows for high resolution in areas of interest, such as coastal areas and areas near boundary conditions, while the computational time is decreased by lower resolution in other areas. For the vertical layers of the grid, the σ-layer approach is used, see section 3.5.1. For the DCSM model, 20 σ-layers with a uniform thickness of 5% of the local water depth are applied.
At the open ended water boundaries at the northern, western and southern side, waterlevel boundary conditions are applied with astronomical waterlevels obtained from the harmonic expansion of 32 tidal

constituents. Temperature boundary conditions are obtained from a dataset World Ocean Atlas 2013 [38] and temperature is modelled using heat-flux models. A discharge is present through 10 incoming rivers in the Netherlands and Germany. The model has been validated with measurements from measuring stations along the coastline, see Figure 8.2. The rmse is about 9 to 10 cm for waterlevel and 0.7 °C for temperature [27], while a more detailed validation is presented in [88].

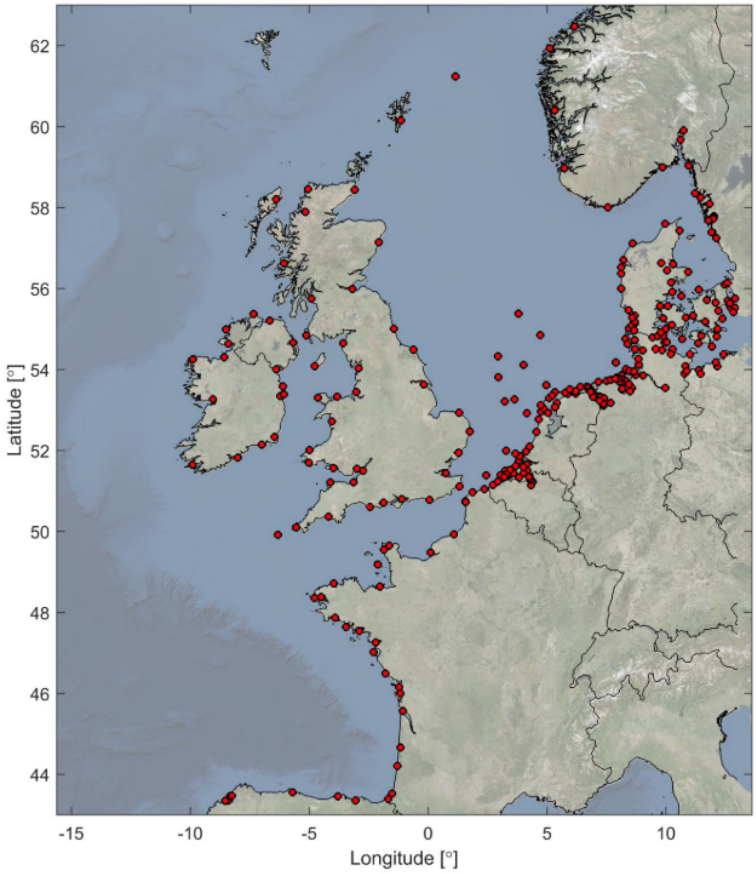


Figure 8.2: The measurement stations present at the DCSM model and used for validation of the DCSM model [85]. Figure retrieved from [85].

The high-resolutional model covering a large area has its disadvantages. The computational time to progress this model is 13 minutes per simulation-day, which means 3,3 days per simulation-year. Therefore, running experiments with this model is quite time consuming.
The Belgian pilot lies within the red coastal area in Figure 8.1, with a resolution (grid size) of approximately 0.5nm (nautical miles, $1\text{nm} = 1.85\text{km}$). To reduce the computational time, a subset of the DCSM grid can be taken where redundant areas will be neglected, but all important tidal effects are still present.
Applying data assimilation and the Ensemble Kalman filter will be the next step for producing a high-resolutional forecasting system for the Belgian pilot in the UNITED project. With slight adjustments in OpenDA, it should be possible to assimilate water temperature for the DCSM model as performed with the 2D model in subsection 7.3.4. On this scale, localization as addition to the Ensemble Kalman filter, as mentioned in section 1.3, can be applied. It will assure that the effect of assimilation will be present only in the local proximity of the measuring stations. It is beyond the scope of this thesis to go into much more detail to test this model.

9 Conclusions

This thesis aims to show the effect of the data assimilation algorithm the Ensemble Kalman filter on estimates of waterlevel, water velocity and temperature in models for the Western Scheldt estuary. A 1D model and a 2D model are investigated, where twin experiments provided the basis for comparison between the models without and with assimilation of generated observations. The 1D model has been implemented in Python and provided for understanding the Ensemble Kalman filter. The software packages Delft3D FM and OpenDA are used for the implementation of the 2D model. These software packages will be needed in the future for a high-resolutional forecasting system using the DCSM model for monitoring and operations for the Belgian pilot of the UNITED project.
The research questions posed in section 1.4 provided the main goals for this thesis and are answered by the conclusions found by experimenting with the two different models examined in this thesis.

- How much does the implementation of the Ensemble Kalman filter on the hydrodynamic models improve the prediction of waterlevel, water velocity and temperature in comparison to the prediction of only the hydrodynamic model?

  The estimate accuracy depends on the model settings and settings of the Ensemble Kalman filter. The Ensemble Kalman filter applied on the 1D model with assimilation of waterlevel and water velocity showed improvement in all cases where observations close to the model predictions were used. The magnitude of improvement varies with different settings for the model and EnKF parameters, in the best case an improvement of four times the accuracy, in lesser cases only slight improvement. For the 2D model assimilation of waterlevel and water velocity observations resulted in improved estimates, where assimilation of temperature resulted in worse estimates.

- Assimilation of which physical variable gives the greatest reduction in prediction errors?

  The assimilation of waterlevel gives the greatest reduction in prediction errors. For the 1D model, assimilation of waterlevel had the most effect on decreasing the rmse for the estimation. Assimilation of water velocity lowered the rmse, but assimilation of only waterlevel at the same location offered better estimations.

- What is the influence of assimilation applied at different locations on the prediction?

  The influence of assimilation is location dependent. Assimilation locations close to or at Cadzand in the 1D model have proven to give the most accurate estimates. In the 2D model, assimilation at the estimates for Station01 improve most by assimilation of observations at Station02 and Station03. Estimates for Station02 improve most by assimilation of observations at Station03. Since both Cadzand and Station03 are located close to the tidal boundary condition with added noise, the hypothesis is that assimilation at locations close to boundary conditions with introduced noise is most beneficial. However, providing definitive proof for this was not the aim of this thesis.

- How much does the estimation through the water column in a 2D model with depth improve by assimilating only observations at the surface for e.g. water velocity or temperature?

  For water velocity a decrease in rmse for assimilated model of two times the initial rmse for the model was discovered for some of the observed locations. Temperature assimilation was executed but proved to give worse estimates than the model itself. A possible explanation for this may be the absence of constant model error covariance, causing the Kalman Gain in the Ensemble Kalman filter to change quickly, which could cause rapid changes in estimates. Furthermore, a collapse of the Ensemble Kalman filter could be occuring due to small standard deviation of the estimation. This may be resolved by restarting the algorithm before the filter collapses, by creation of a new ensemble at this time.

Based on this research, multiple ways to go forward could be investigated. Firstly, we recommend researching if temperature assimilation in a different model setup with more realistic longer spin-up time of temperature, will give rise to better estimate accuracy within the water column. Then, a collapsing Ensemble Kalman filter can be prevented by initializing a new ensemble just before the collapse occurs. Furthermore, assimilation of other physical variables, such as salinity and suspended particulate matter, is advised to investigate to see if assimilation on the surface will result in better estimation within the water column. The answer to these questions will determine whether assimilation of surface observations will improve forecasts within the water column sufficiently or not, for physical variables important for cultivation of seaweed and flat oysters.
Although with slight adjustments in OpenDA, it should be possible to assimilate water temperature for the DCSM model as we have provided a first proof of concept with the 2D model of the Western Scheldt implemented in Delft3D FM. Experiments to see where assimilation is best to apply and possibly apply localization to reduce the effect of assimilation to only the local proximity of the assimilation locations. Other interesting questions may be to look at how far in the future the model with assimilation still gives accurate predictions.

A

Derivations model error covariance

In this appendix, a result needed for derivation of the optimal Kalman Gain in section 4.2 is derived:

$$\mathbf{P}_k^a \equiv \mathrm{Cov}(\mathbf{x}_k - \mathbf{x}_k^a) = \mathbf{P}_k^f - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^f - \mathbf{P}_k^f\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)\mathbf{K}_k^T. \quad \text{(A.1)}$$

Using the definitions of $\mathbf{P}_k^f = \mathrm{Cov}(\mathbf{x}_k - \mathbf{x}_k^f)$, $\mathbf{P}_k^a = \mathrm{Cov}(\mathbf{x}_k - \mathbf{x}_k^a)$, $\mathbf{x}_k^a = \mathbf{x}_k^f + \mathbf{K}_k(\mathbf{z}^k - \mathbf{H}_k\mathbf{x}_k^f)$, $\mathbf{R}_k = \mathrm{Cov}(\mathbf{v}_k)$ and $\boldsymbol{z}_k = \mathbf{H}_k\mathbf{x} + \mathbf{v}_k$ and covariance properties:

$$\begin{aligned}
&\mathbf{P}_k^a \equiv \mathrm{Cov}(\mathbf{x}_k - \mathbf{x}_k^a) && \Leftrightarrow\\
&\mathbf{P}_k^a = \mathrm{Cov}[\mathbf{x}_k - (\mathbf{x}_k^f + \mathbf{K}_k(\boldsymbol{z}_k - \mathbf{H}_k\mathbf{x}_k^f))] && \Leftrightarrow\\
&\mathbf{P}_k^a = \mathrm{Cov}(\mathbf{x}_k - [\mathbf{x}_k^f + \mathbf{K}_k(\mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k - \mathbf{H}_k\mathbf{x}_k^f)]) && \Leftrightarrow\\
&\mathbf{P}_k^a = \mathrm{Cov}[(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)(\mathbf{x}_k - \mathbf{x}_k^f) - \mathbf{K}_k\mathbf{v}_k] && \Leftrightarrow\\
&\mathbf{P}_k^a = \mathrm{Cov}[(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)(\mathbf{x}_k - \mathbf{x}_k^f)] + \mathrm{Cov}[\mathbf{K}_k\mathbf{v}_k] && \Leftrightarrow\\
&\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathrm{Cov}(\mathbf{x}_k - \mathbf{x}_k^f)(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathrm{Cov}(\mathbf{v}_k)\mathbf{K}^T && \Leftrightarrow\\
&\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^f(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}^T && \Leftrightarrow\\
&\mathbf{P}_k^a = \mathbf{P}_k^f - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^f - \mathbf{P}_k^f\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)\mathbf{K}_k^T.
\end{aligned}$$

Furthermore, the formulation of the analysis model error covariance $\mathbf{P}_k^a$, used in the Kalman filter in algorithm 3 is derived, using Equation A.1 and the optimal Kalman Gain $\mathbf{K}_k = \mathbf{P}_k^f\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$ (Equation 4.5):

$$\begin{aligned}
&\mathbf{P}_k^a = \mathbf{P}_k^f - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^f - \mathbf{P}_k^f\mathbf{H}_k^T\mathbf{K}_k^T + \mathbf{K}_k(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)\mathbf{K}_k^T && \Leftrightarrow\\
&\mathbf{P}_k^a = \mathbf{P}_k^f - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^f - \mathbf{P}_k^f\mathbf{H}_k^T\mathbf{K}_k^T + (\mathbf{P}_k^f\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)^{-1})(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k)\mathbf{K}_k^T && \Leftrightarrow\\
&\mathbf{P}_k^a = \mathbf{P}_k^f - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^f && \Leftrightarrow\\
&\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^f,
\end{aligned}$$

and therefore:

$$\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^f. \quad \text{(A.2)}$$

B

Additional estimates 1D model experiment 3

EnKF h Cad

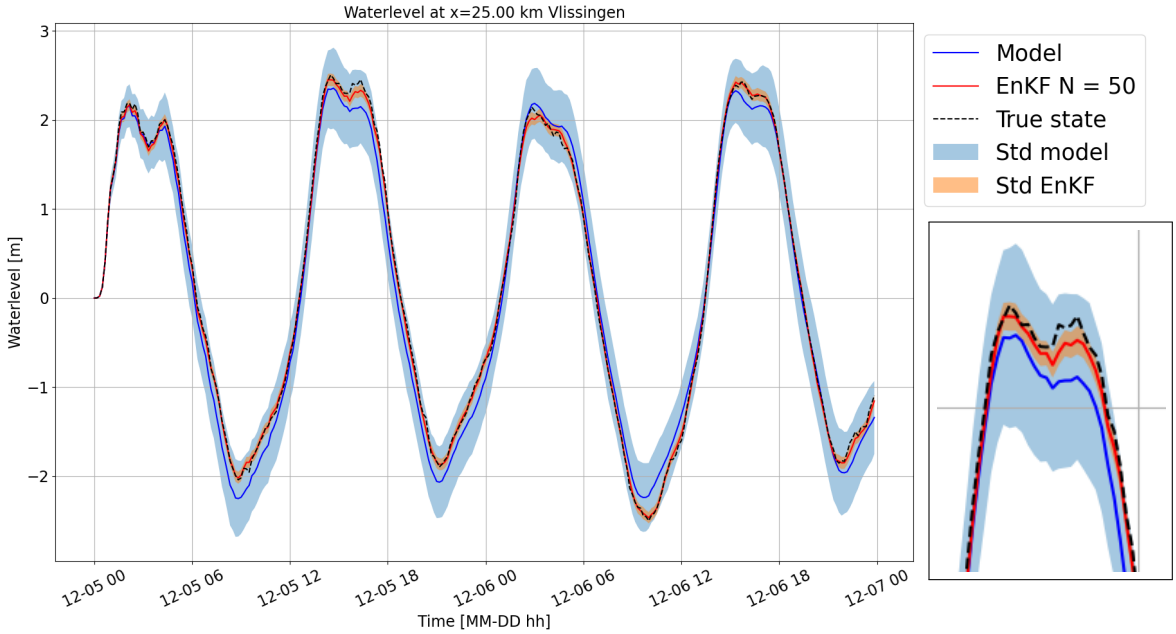


Figure B.1: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF h Cad with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF h Cad. Yellow: standard deviation of EnKF h Cad. Black striped: chosen true state from the model. Black dots: generated measurements.

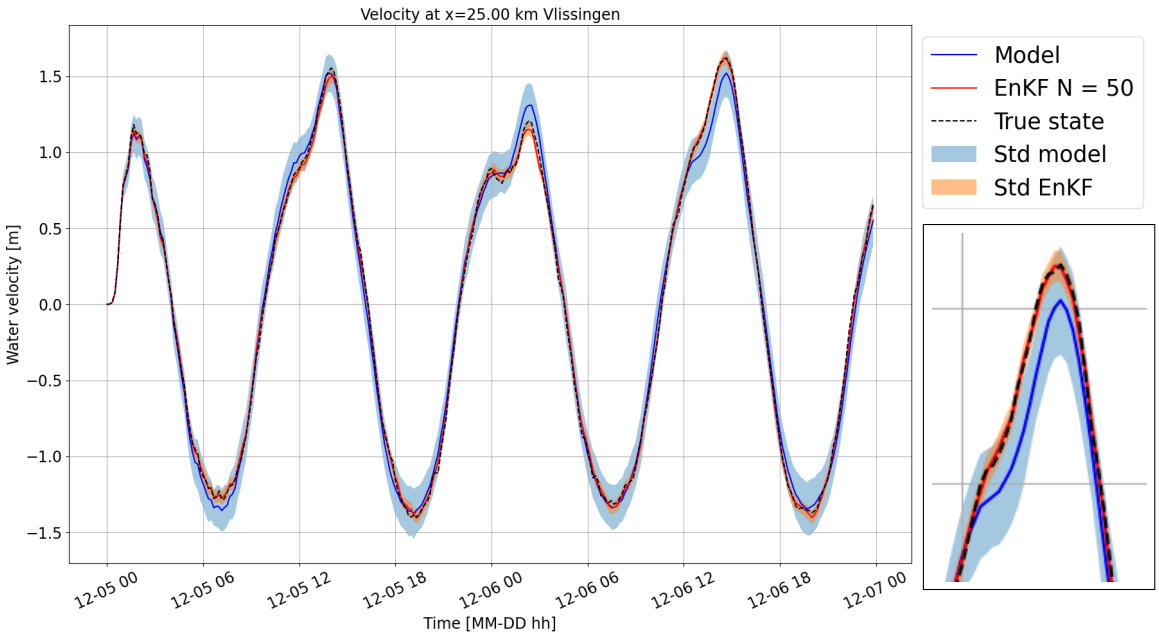


Figure B.2: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF h Cad with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF h Cad. Yellow: standard deviation of EnKF h Cad. Black striped: chosen true state from the model.

EnKF h Ba

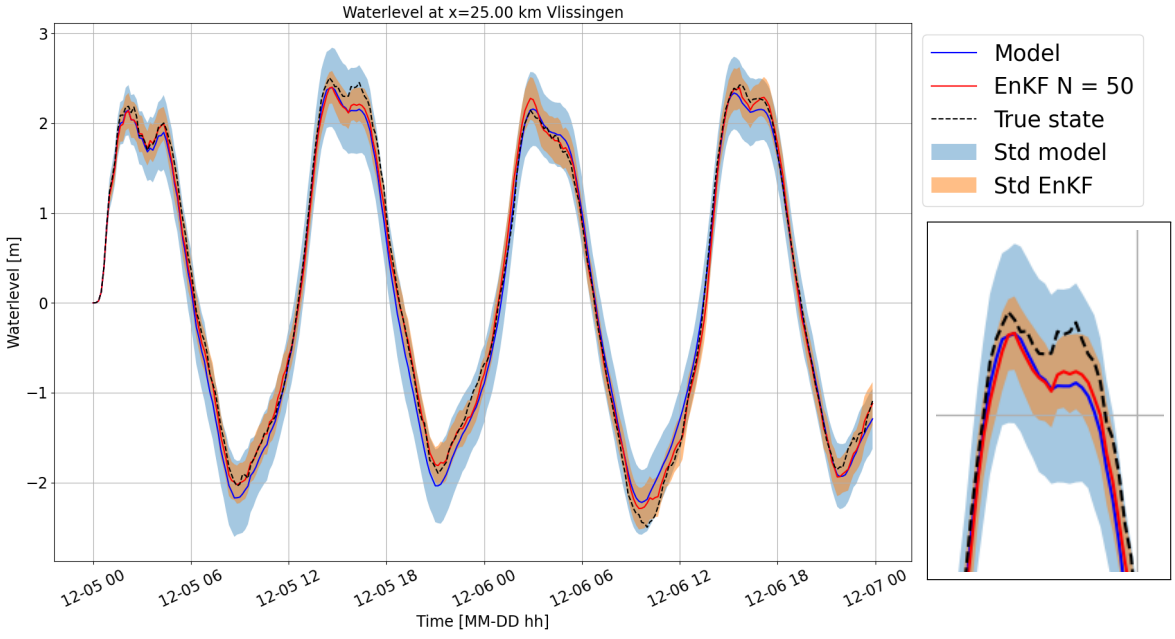


Figure B.3: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF h Ba with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF h Ba. Yellow: standard deviation of EnKF h Ba. Black striped: chosen true state from the model. Black dots: generated measurements.

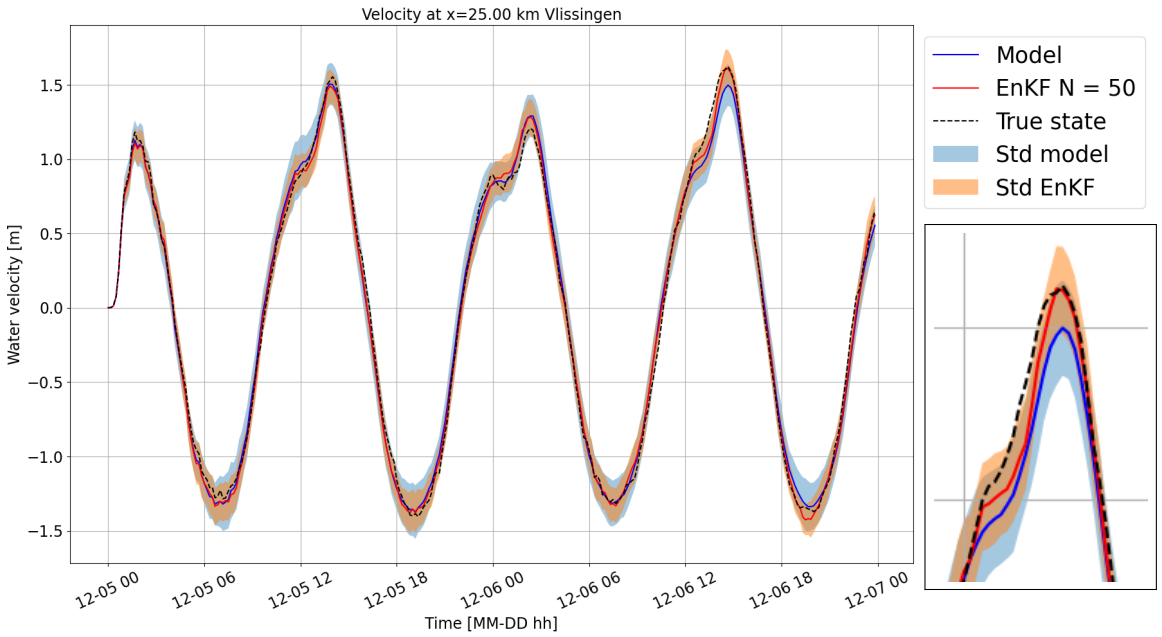


Figure B.4: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF h Ba with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF h Ba. Yellow: standard deviation of EnKF h Ba. Black striped: chosen true state from the model.

EnKF u Cad

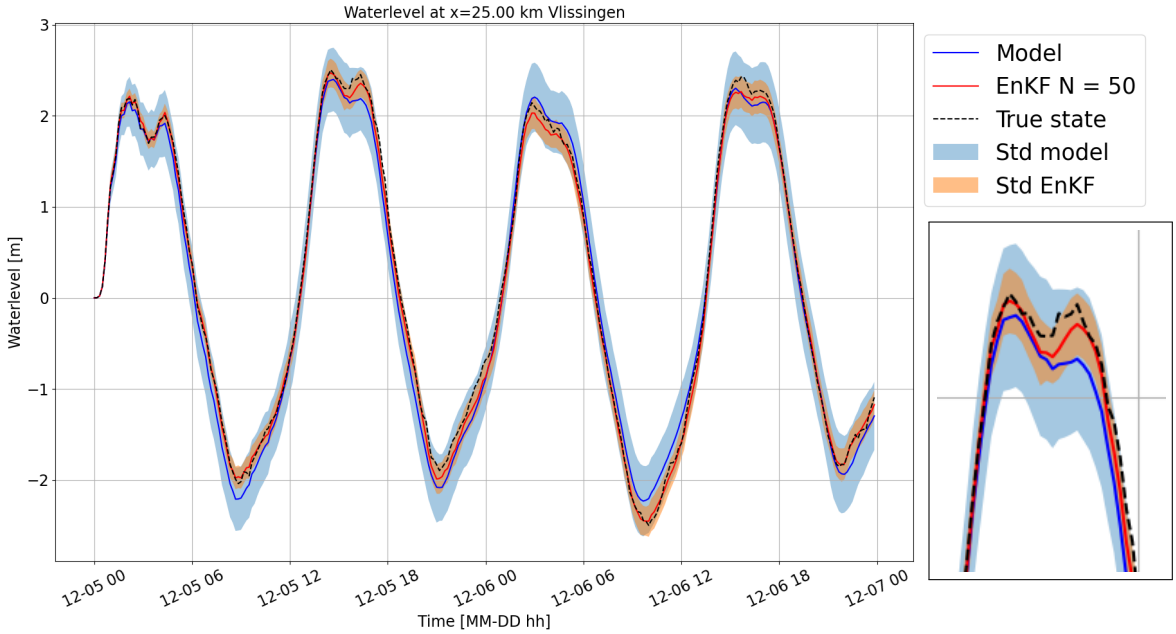


Figure B.5: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF u Cad with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF u Cad. Yellow: standard deviation of EnKF u Cad. Black striped: chosen true state from the model. Black dots: generated measurements.

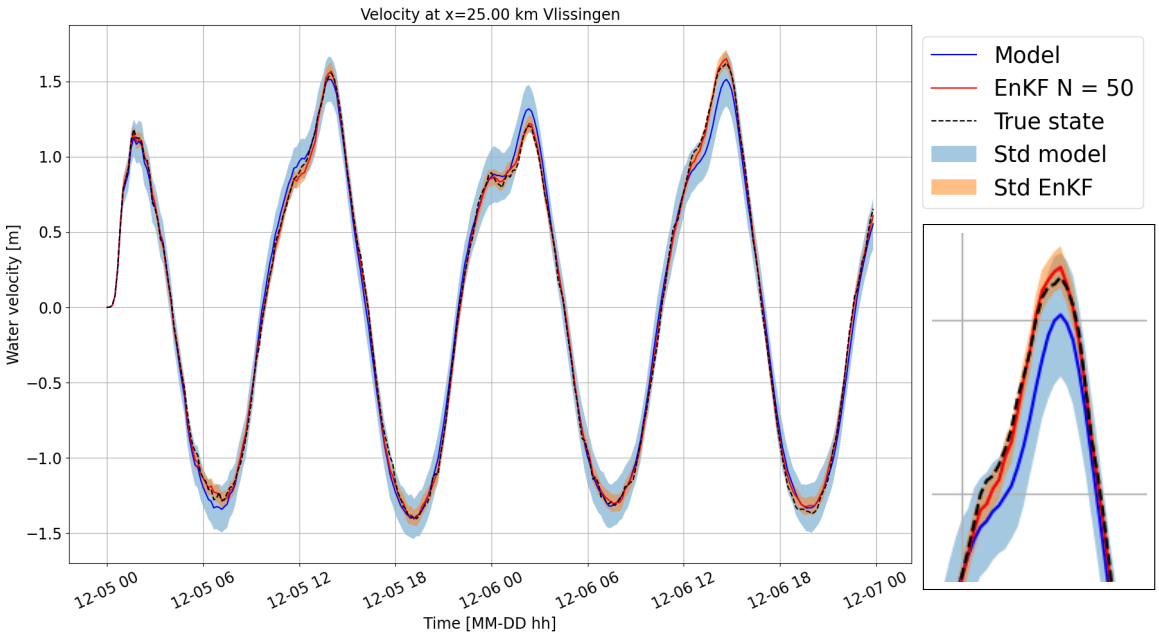


Figure B.6: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF u Cad with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF u Cad. Yellow: standard deviation of EnKF u Cad. Black striped: chosen true state from the model.

EnKF u Ha

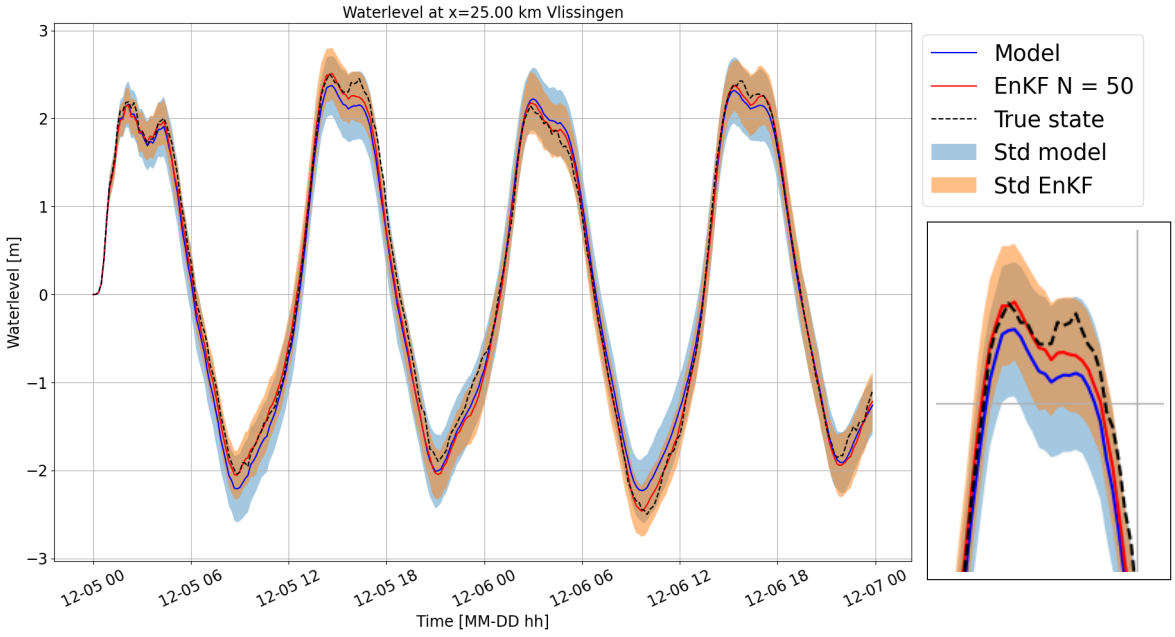


Figure B.7: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF u Ha with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF u Ha. Yellow: standard deviation of EnKF u Ha. Black striped: chosen true state from the model. Black dots: generated measurements.

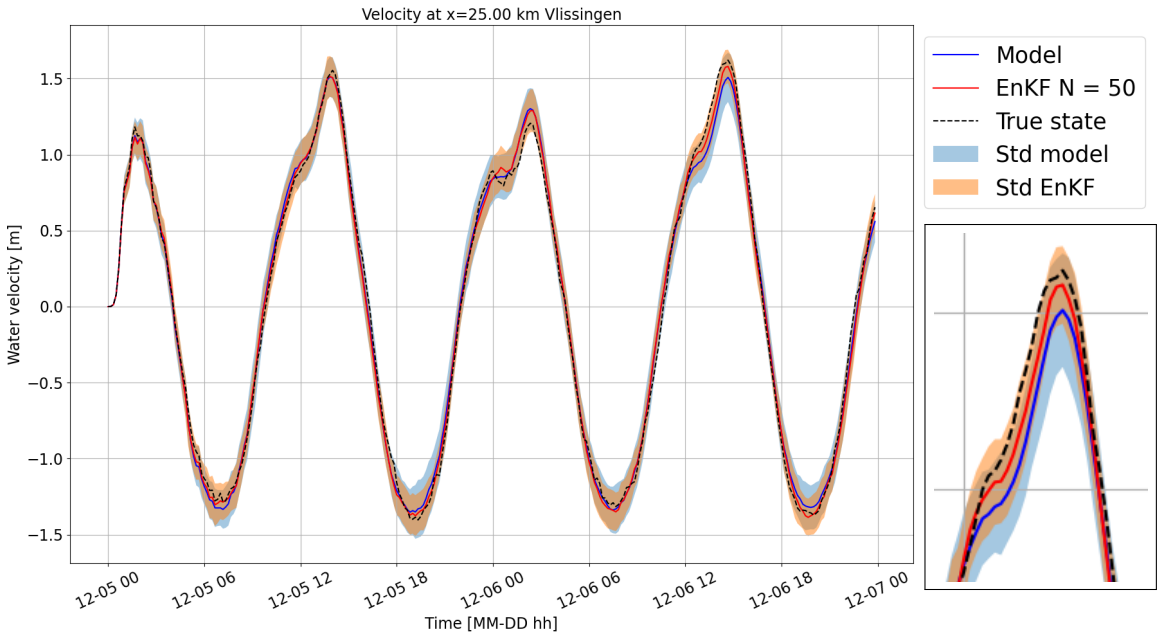


Figure B.8: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and EnKF u Ha with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF u Ha. Yellow: standard deviation of EnKF u Ha. Black striped: chosen true state from the model.

C

Additional estimates 1D model experiment 4

$T = 2\Delta t$

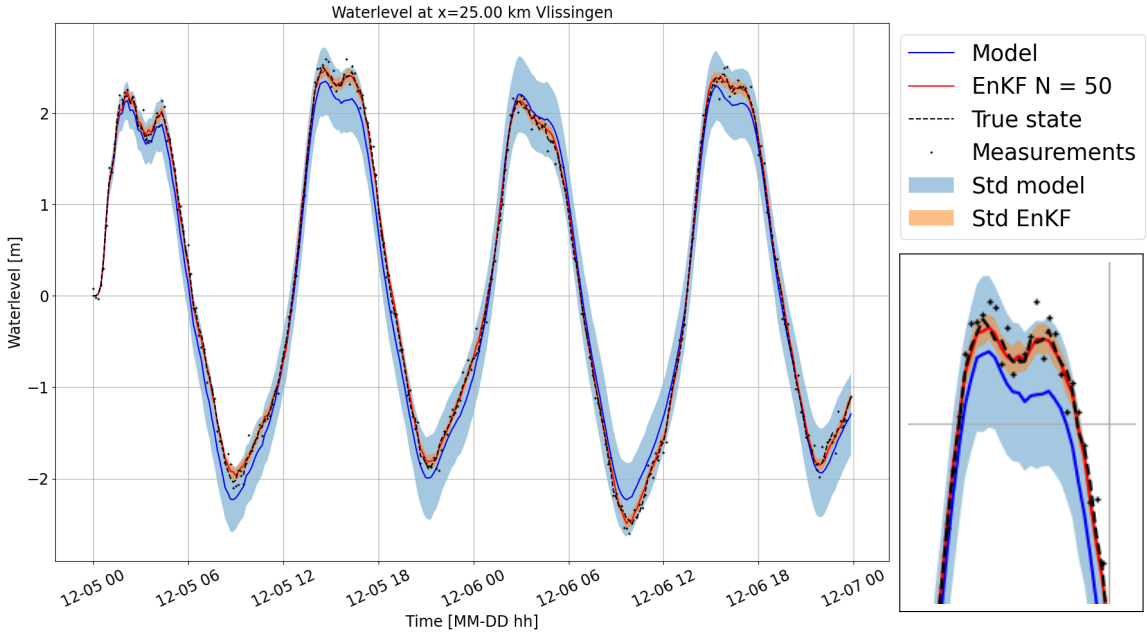


Figure C.1: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF5 with assimilation frequency of $T = 2\Delta t$ and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model. Black dots: measurements.

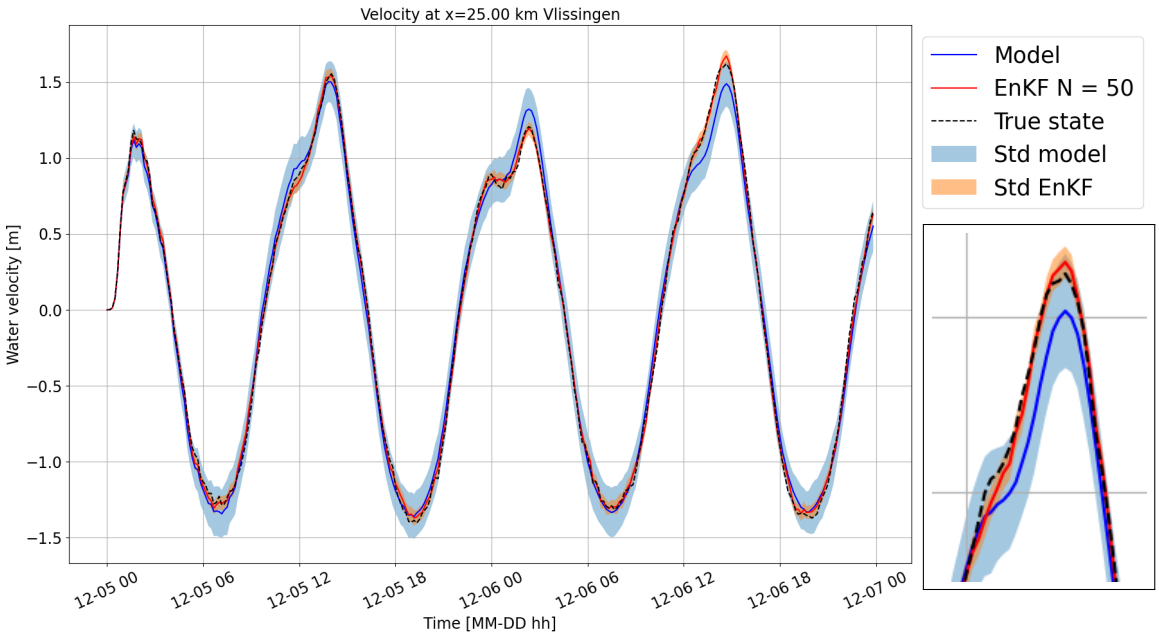


Figure C.2: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF5 with assimilation frequency of $T = 2\Delta t$ and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model.

$T = 10\Delta t$

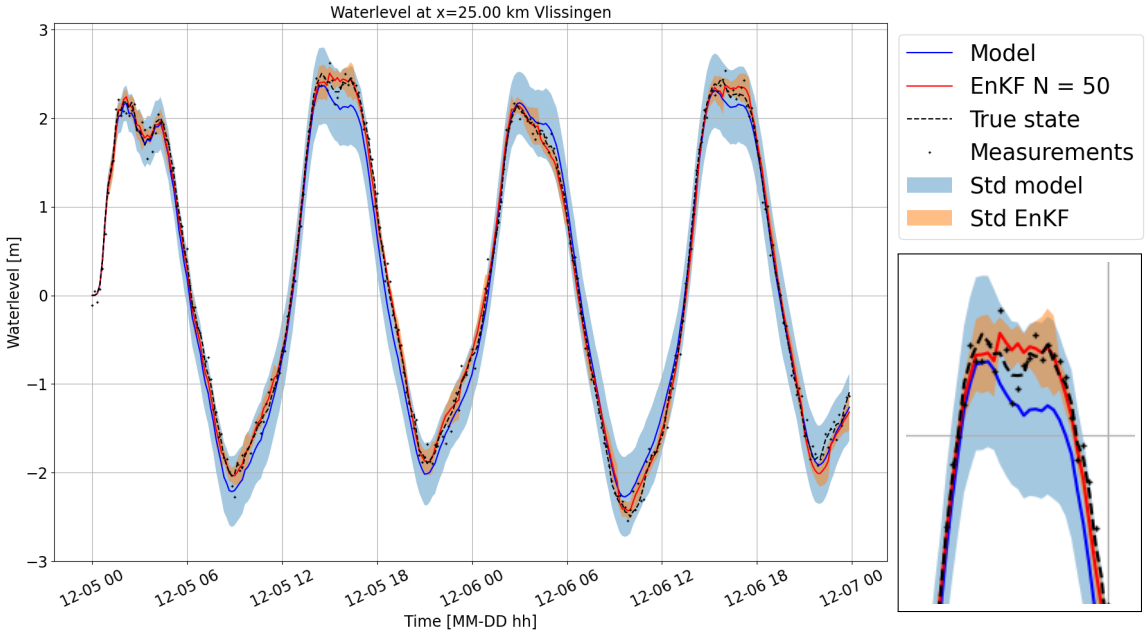


Figure C.3: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF5 with assimilation frequency of $T = 10\Delta t$ and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model. Black dots: measurements.

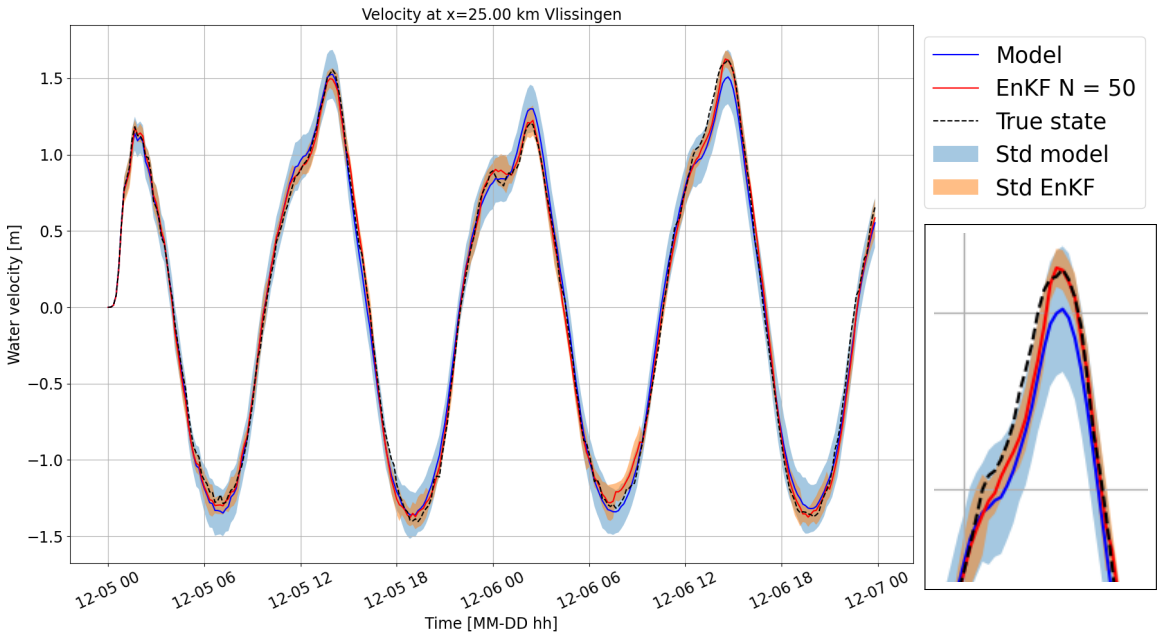


Figure C.4: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF5 with assimilation frequency of $T = 10\Delta t$ and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model.

$T = 40\Delta t$

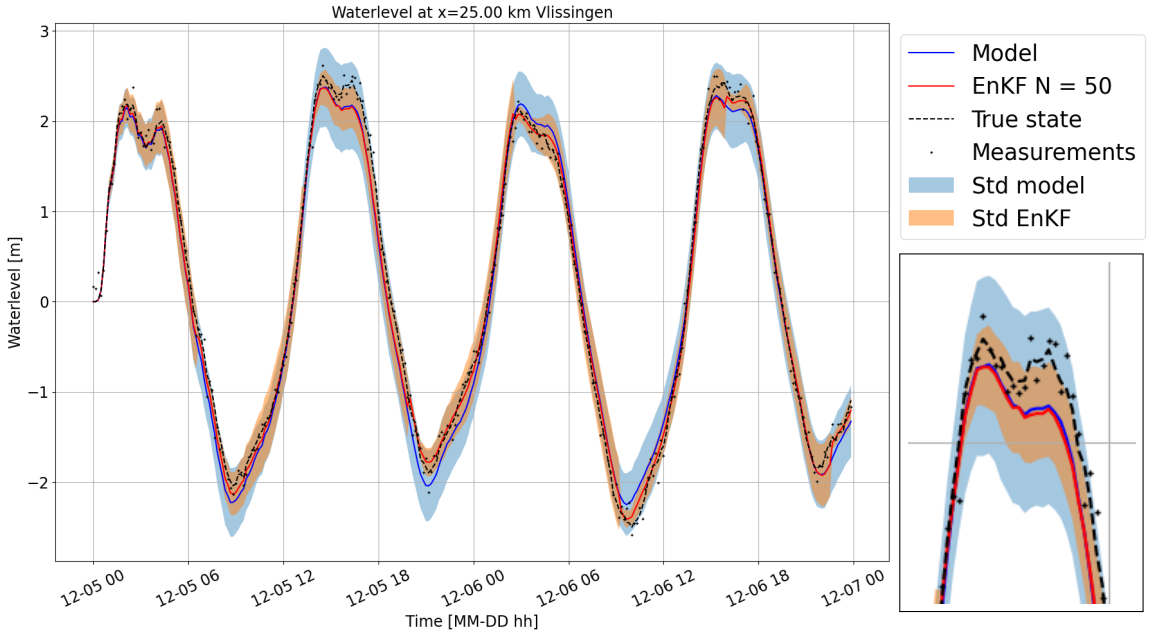


Figure C.5: The waterlevel estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF5 with assimilation frequency of $T = 40\Delta t$ and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 5th of December 12:00 and 5th of December 19:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model. Black dots: measurements.

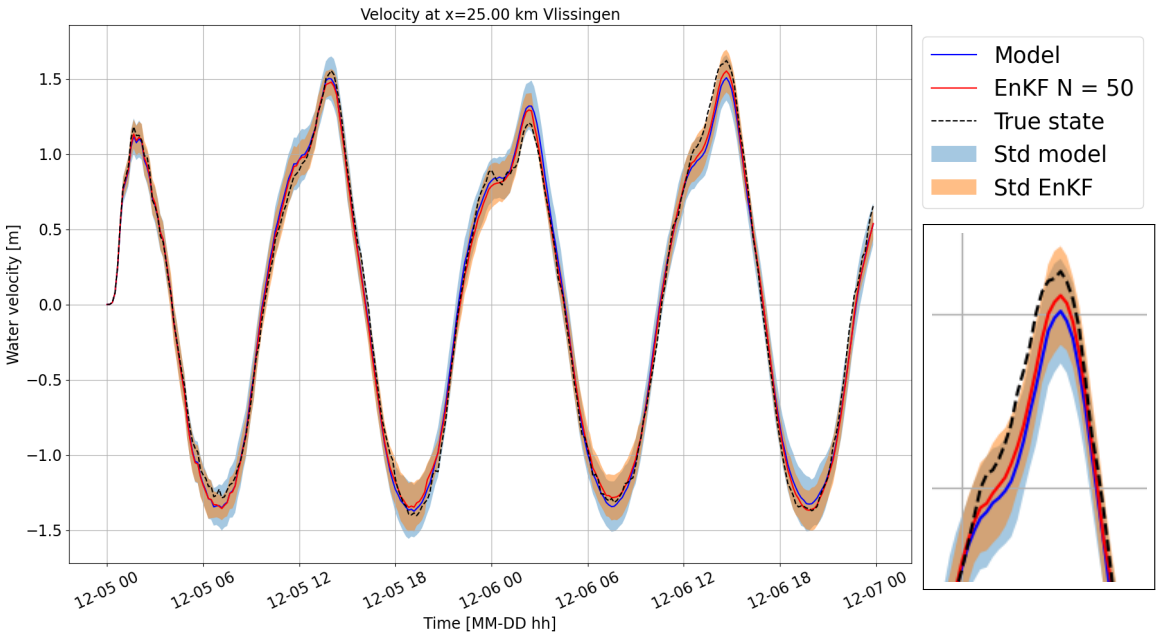


Figure C.6: The velocity estimates at Vlissingen over two days, computed with the hydrodynamic model and the EnKF5 with assimilation frequency of $T = 40\Delta t$ and furthermore with settings in Table 6.1. Left: over two days from 5th of December until 7th of December. Right: Zoomed in peak between 6th of December 10:00 and 6th of December 17:00. Blue: The estimate of the model. Light blue: standard deviation of the model estimate. Red: Estimate of EnKF5. Yellow: standard deviation of EnKF5. Black striped: chosen true state from the model.

D

Additional deterministic estimates 2D model

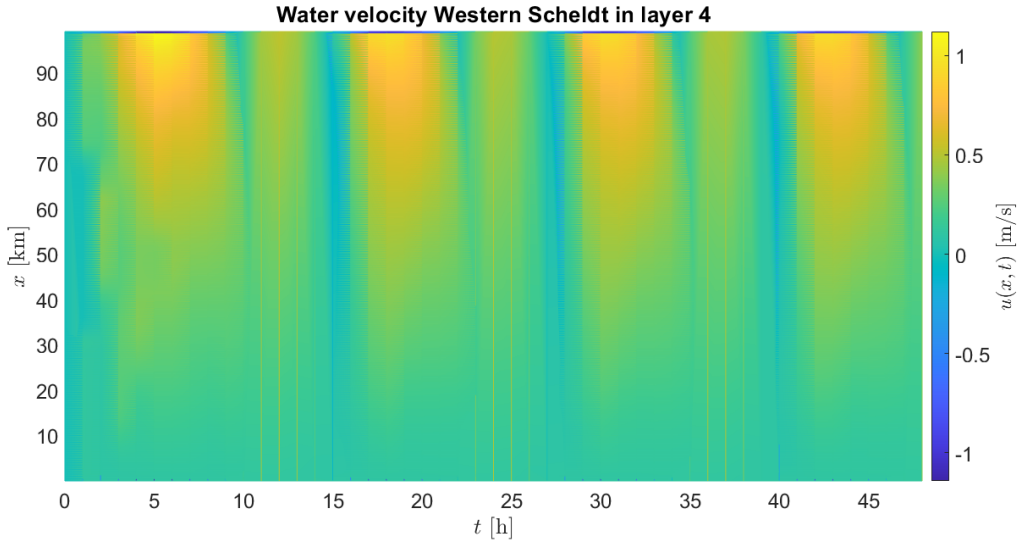


Figure D.1: Deterministic estimate of water velocity $u(x,t)$ at σ-layer 4 of the 2D estuary model, plotted as a function of position x and time t after 1991-01-01 00:00.

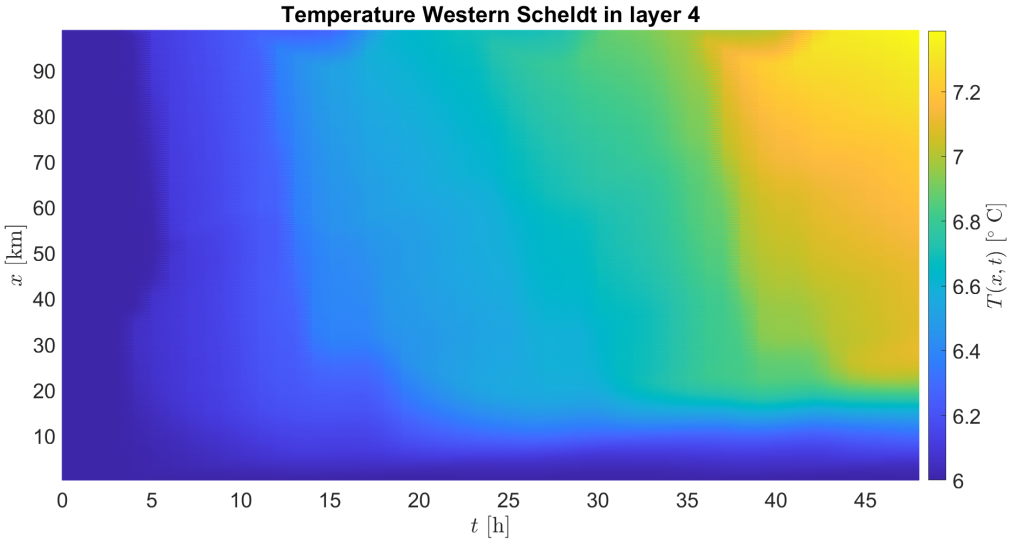


Figure D.2: Deterministic estimate of temperature $T(x,t)$ at σ-layer 4 of the 2D estuary model, plotted as a function of position x and time t after 1991-01-01 00:00.

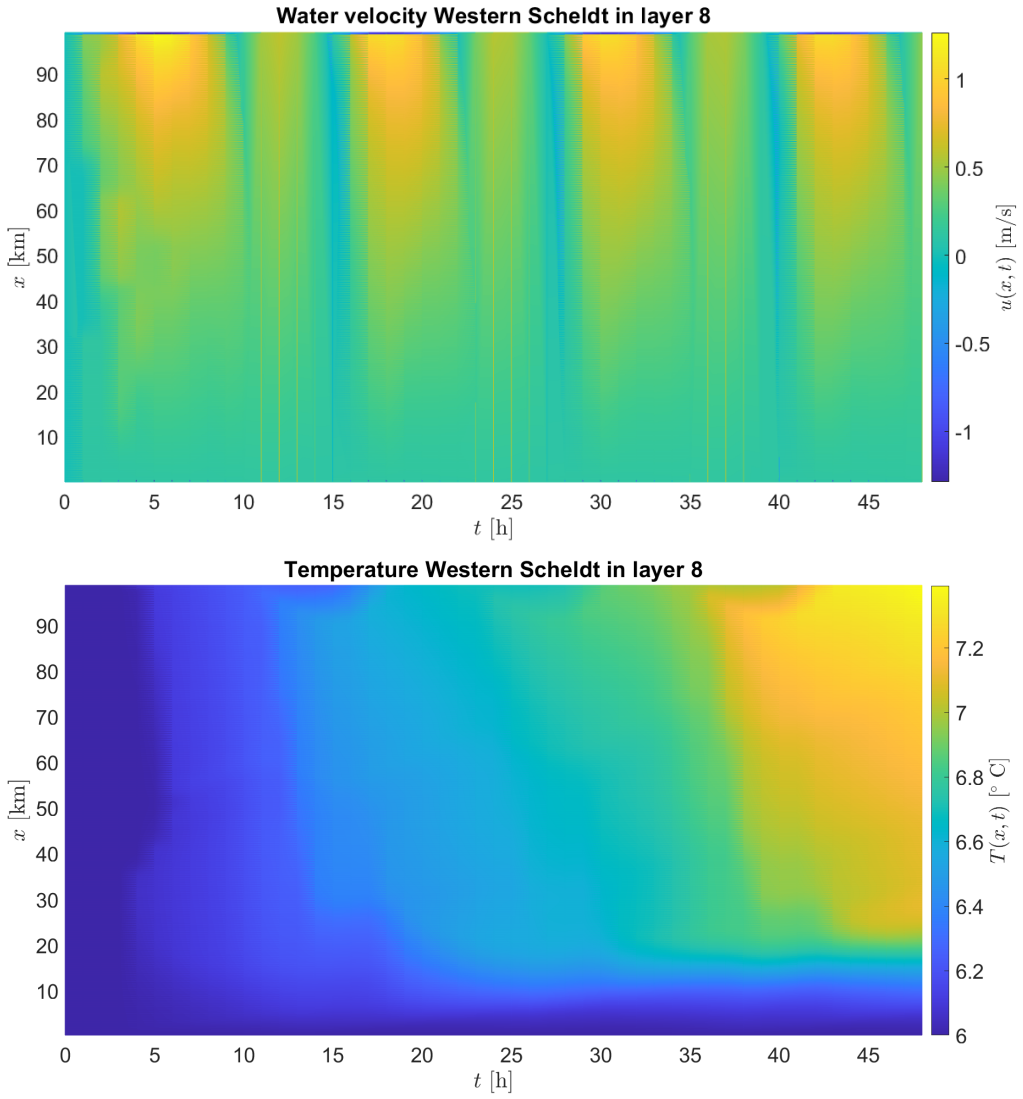


Figure D.3: Deterministic estimates of the 2D estuary model, plotted as a function of position x and time t after 1991-01-01 00:00. Top: Velocity $u(x,t)$ at σ-layer 8. Bottom: Temperature $T(x,t)$ at σ-layer 8.

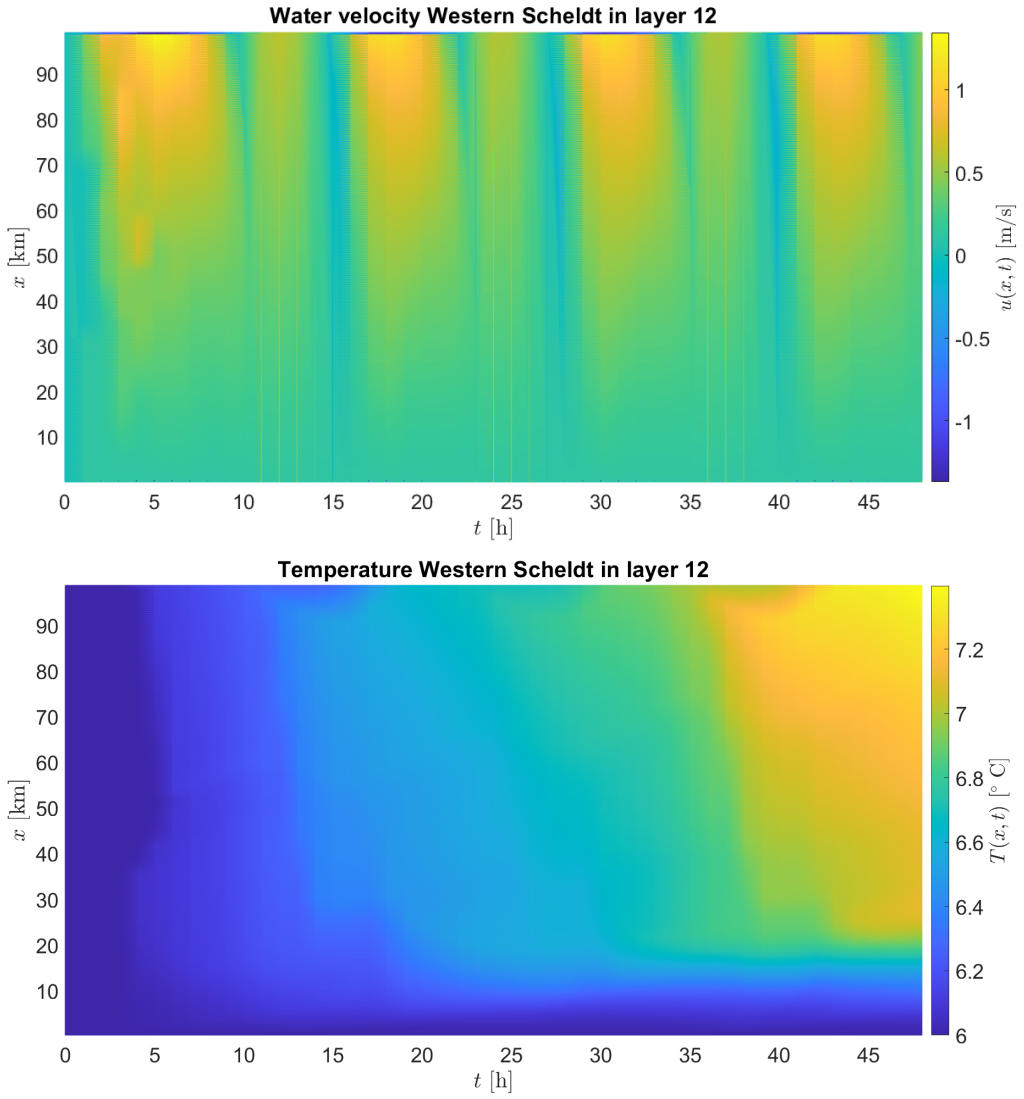


Figure D.4: Deterministic estimates of the 2D estuary model, plotted as a function of position x and time t after 1991-01-01 00:00. Top: Velocity $u(x,t)$ at σ-layer 12. Bottom: Temperature $T(x,t)$ at σ-layer 12.

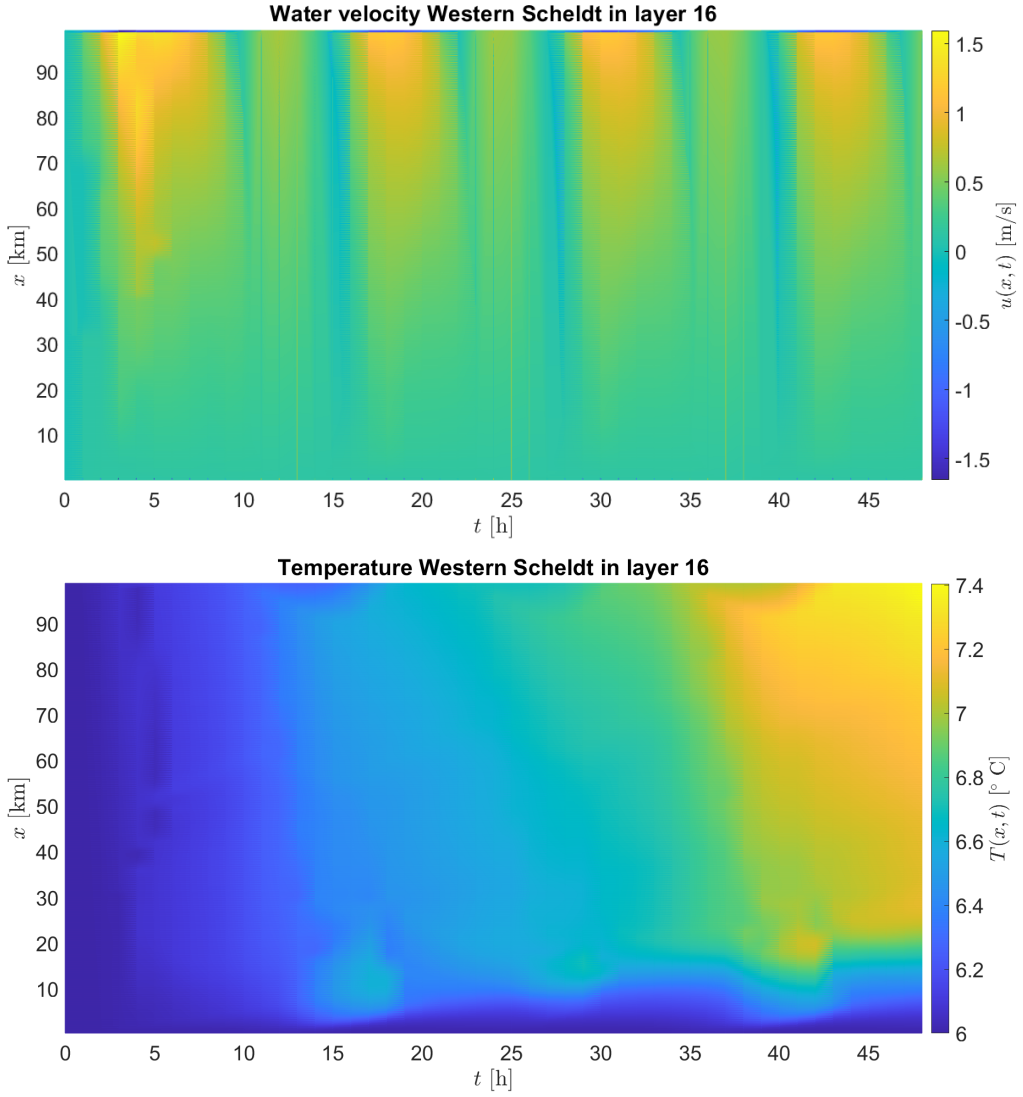


Figure D.5: Deterministic estimates of the 2D estuary model, plotted as a function of position x and time t after 1991-01-01 00:00. Top: Velocity $u(x,t)$ at σ-layer 16. Bottom: Temperature $T(x,t)$ at σ-layer 16.

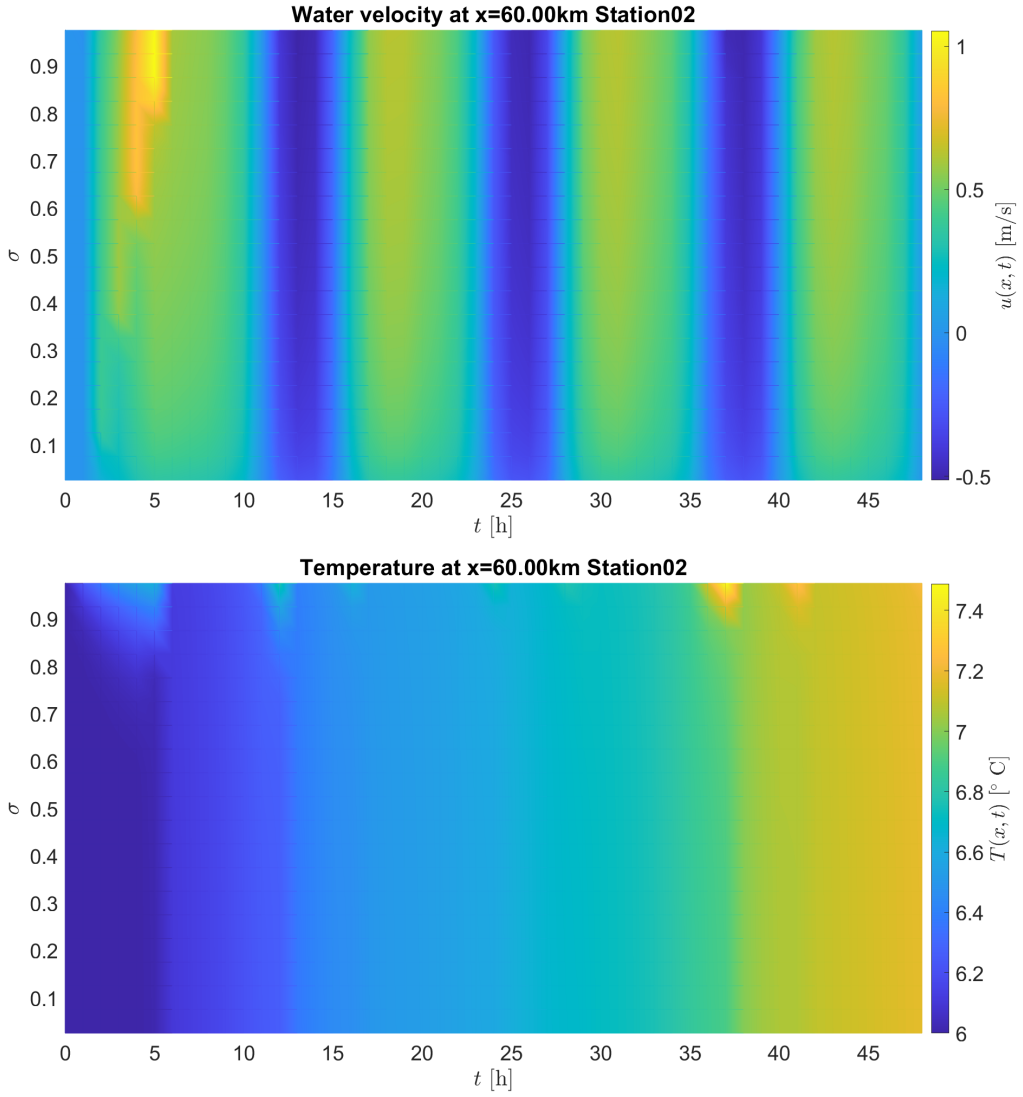


Figure D.6: Deterministic estimate of the depth profile of velocity $u(x,t)$ and temperature $T(x,t)$ at Station02 over two days after 1991-01-01 00:00, plotted as a function of time t and σ-layers, where 1 is the water surface and 0 the bed level divided in 20 layers.

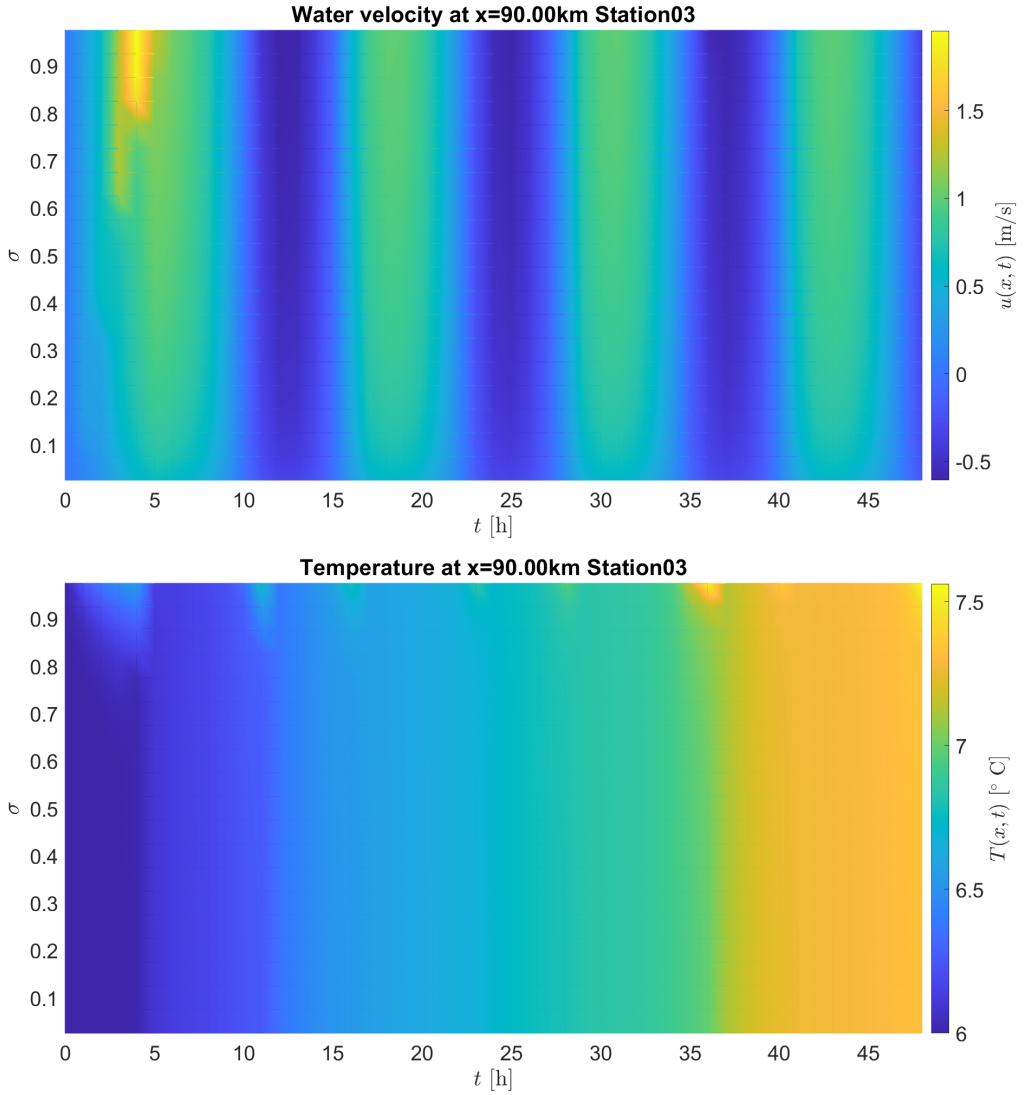


Figure D.7: Deterministic estimate of the depth profile of velocity $u(x,t)$ and temperature $T(x,t)$ at Station03 over two days after 1991-01-01 00:00, plotted as a function of time t and σ-layers, where 1 is the water surface and 0 the bed level divided in 20 layers.

Bibliography

[1] The data assimilation research testbed. https://dart.ucar.edu/. Online accessed on 2021-06-30.

[2] Deltares. https://www.deltares.nl/en/. Online accessed on 2021-06-24.

[3] PDAF: Parallel Data Assimilation Framework. https://pdaf.awi.de/trac/wiki. Online accessed on 2021-06-30.

[4] Sequoia data assimilation platform. https://sirocco.obs-mip.fr/assimilation-tools/sequoia-data-assimilation-platform/. Online accessed on 2021-06-30.

[5] OpenDA. https://www.openda.org/. Online accessed on 2021-06-24.

[6] Parkwind, offshore green energy. https://parkwind.be/about-us/, 2019. Online accessed on 2021-07-06.

[7] United. https://www.northseafarmers.org/projects/united, 2020. Online accessed on 2021-07-01.

[8] United: Offshore wind, flat oyster aquaculture & restoration, & seaweed cultivation in Belgium; where aquaculture feeds ecosystem restoration. https://www.h2020united.eu/pilots/2-uncategorised/42-offshore-wind-and-flat-oyster-aquaculture-restoration-in-belgium, 2020. Online accessed on 2021-07-06.

[9] United. https://www.h2020united.eu/pilots, 2020. Online accessed on 2020-09-16.

[10] United: Offshore seaweed and floating solar energy in the Netherlands. https://www.h2020united.eu/pilots/2-uncategorised/47-offshore-seaweed-and-floating-solar-energy-in-the-netherlands, 2020. Online accessed on 2020-09-16.

[11] What is data assimilation? https://research.reading.ac.uk/met-darc/aboutus/what-is-data-assimilation/#:~:text=Data%20assimilation%20is%20the%20science,call%20this%20state%20an%20analysis., 2020. Online accessed on 2021-04-28.

[12] What causes algal blooms? https://cees.iupui.edu/research/algal-toxicology/bloomfactors, 2021. Online accessed on 2021-07-19.

[13] Theo Baracchini, Philip Y. Chu, Jonas Sukys, Gian Lieberherr, Stefan Wunderle, Alfred Wüest, and Damien Bouffard. Data assimilation of in situ and satellite remote sensing data to 3D hydrodynamic lake models: a case study using Delft3D-FLOW v4.03 and OpenDA v2.4. *Geoscientific Model Development*, 13:1267–1284, 2020. doi: 10.5194/gmd-13-1267-2020.

[14] Bert Bolin. Carl-Gustaf Rossby The Stockholm period 1947–1957. *Tellus A: Dynamic Meteorology and Oceanography*, 51(1):4–12, 1999. doi: 10.3402/tellusa.v51i1.12285.

[15] Eef Brouwers. Haal meer uit je windpark: aan de slag met medegebruik. https://www.northseafarmers.org/news/2020/Artikel-WindNieuws-MU-Procedure_12May20.pdf, 2020. Online accessed on 2021-07-01.

[16] Bela Buck, Michael Ebeling, and Tanja Michler-Cieluch. Mussel cultivation as a co-use in offshore wind farms: Potential and economic feasibility. *Aquaculture Economics & Management*, 14:255–281, 10 2010. doi: 10.1080/13657305.2010.526018.

[17] Gerrit Burgers, Peter Jan Van Leeuwen, and Geir Evensen. On the analysis scheme in the ensemble kalman filter. *Monthly Weather Review*, 126:1719–1724, June 1998. doi: 10.1175/1520-0493(1998)126<1719:ASITEK>2.0.CO;2.

[18] ESM: Advanced Earth System Modelling Capacity. Data assimilation with the parallel data assimilation framework: A short course at egu19, 2019. https://www.esm-project.net/news/news/detail/News/data-assimilation-with-the-parallel-data-assimilation-framework-a-short-course-at-egu19/.

[19] Andrew Chadwick. Shallow-water wave theory. http://www.coastalwiki.org/wiki/Shallow-water_wave_theory#Water_particle_velocities.2C_accelerations_and_paths, 2021. Online accessed on 2021-8-20.

[20] J. G. Charney, R. Fjörtoft, and J. Von Neumann. Numerical integration of the barotropic vorticity equation. *A Quarterly Journal of Geophysics*, 2(4), 1950. doi: 10.3402/tellusa.v2i4.8607.

[21] Bérenger Colsoul, Pierre Boudry, María Luz Pérez-Parallé, Ana Bratoš Cetinić, Tristan Hugh-Jones, Isabelle Arzul, Nicolas Mérou, Karl Mathias Wegner, Corina Peter, Verena Merk, and Bernadette Pogoda. Sustainable large-scale production of European flat oyster (Ostrea edulis) seed for ecological restoration and aquaculture: a review. *Reviews in Aquaculture*, 13:1423–1468, 2021. doi: 10.1111/raq.12529.

[22] Erik D. de Goede. Delft3D FM - Coastal Hydrodynamic Modelling (blended educational experience), DSD-INT 2021. https://www.deltares.nl/en/academy/delft3d-flexible-mesh-coastal-hydrodynamic-modelling-online-course-dsd-int-2021/.

[23] Deltares. Delft3D 4 and Delft3D FM. https://www.deltares.nl/en/software-solutions/delft3d-4-delft3d-fm/, . Online accessed on 2021-06-24.

[24] Deltares. Delft3D Flexible Mesh Suite. https://www.deltares.nl/en/software/delft3d-flexible-mesh-suite/, . Online accessed on 2021-06-24.

[25] Deltares. SOBEK Suite. https://www.deltares.nl/en/software/sobek/, . Online accessed on 2021-06-24.

[26] Deltares. Delft3D-Quickplot User Manual. https://oss.deltares.nl/documents/183920/185723/delft3d-quickplot_user_manual.pdf, 2014.

[27] Deltares. 3D Continental Shelf Model - Flexible Mesh. https://www.deltares.nl/en/projects/3d-dutch-continental-shelf-model-flexible-mesh-2/, 2018. Online accessed on 2021-07-20.

[28] Deltares. D-Flow Flexible Mesh, User Manual. https://content.oss.deltares.nl/delft3d/manuals/D-Flow_FM_User_Manual.pdf, 2020.

[29] Deltares. D-Flow Flexible Mesh Technical Reference Manual. https://content.oss.deltares.nl/delft3d/manuals/D-Flow_FM_Technical_Reference_Manual.pdf, 2020.

[30] Deltares. Delft3D-FLOW User Manual. http://content.oss.deltares.nl/delft3d/manuals/Delft3D-FLOW_User_Manual.pdf, 2020.

[31] Deltares. DIMR, Technical Reference Manual. https://content.oss.deltares.nl/delft3d/manuals/DIMR_Technical_reference_Manual.pdf, 2020.

[32] Carlos M. Duarte, Jiaping Wu, Xi Xiao, Annette Bruhn, and Dorte Krause-Jensen. Can seaweed farming play a role in climate change mitigation and adaptation? *Frontiers in Marine Science*, 4: 1–8, 2017. doi: 10.3389/fmars.2017.00100.

[33] Vincent Enchevin and Pierre de Mey. Horizontal and vertical structure of the representer functions for sea surface measurements in a coastal circulation model. *Journal of Physical Oceanography*, 30:2627–2635, 2000. doi: 10.1175/1520-0485(2000)030<2627:HAVSOT>2.0.CO;2.

[34] Geir Evensen. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99(C5), May 1994. doi: 10.1029/94JC00572.

[35] Geir Evensen. The Ensemble Kalman Filter: theoretical formulation and practical implementation. *Ocean Dynamics*, pages 343–367, May 2003. doi: 10.1007/s10236-003-0036-9.

[36] Geir Evensen. The Ensemble Kalman filter for Combined State and Parameter Estimation; Monte Carlo Techniques for Data Assimilation in large systems. *IEEE Control Systems Magazine*, 29: 83–104, June 2009. doi: 10.1109/MCS.2009.932223.

[37] Geir Evensen and Helge Drange. Data assimilation for coastal zone monitoring and forecasting. In *Operational oceanography*, volume 62 of *Elsevier Oceanography Series*, pages 516–522. Elsevier, 1997.

[38] National Centers for Environmental Information. World ocean atlas. https://www.ncei.noaa.gov/products/world-ocean-atlas. Online accessed on 2021-07-20.

[39] P.A. Francis, A.K. Jithin, A. Chatterjee, K. Chakraborty, A. Paul, B. Balaji, and et al. High-Resolution Operational Ocean Forecast and Reanalysis System for the Indian Ocean. *Bulletin of the American Meteorological Society*, 101:E1340–E1356, 2020. doi: 10.1175/BAMS-D-19-0083.1.

[40] T. M. Hamill, J. S. Whitaker, and C. Snyder. Distance-dependent filtering of background error covariance estimates in an ensemble Kalman filter. *Monthly Weather Review*, 129(11):2776–2790, 2001. ISSN 00270644. doi: 10.1175/1520-0493(2001)129<2776:DDFOBE>2.0.CO;2.

[41] Francis H. Harlow and J. Eddie Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The Physics of Fluids*, 8, 1965. doi: 10.1063/1.1761178.

[42] Vibeke Haugen and Geir Evensen. Assimilation of sla and sst data into an ogcm for the indian ocean. *Ocean Dynamics*, 52:133–151, 07 2002. doi: 10.1007/s10236-002-0014-7.

[43] H.R. Hiester, Matthew Piggott, P.E. Farrell, and P. Allison. Assessment of spurious mixing in adaptive mesh simulations of the two-dimensional lock-exchange. *Ocean Modelling*, 73:30–44, 01 2014. doi: 10.1016/j.ocemod.2013.10.003.

[44] Pieter L. Houtekamer and Herschel L. Mitchell. A sequential ensemble kalman filter for atmospheric data assimilation. *Monthly Weather Review*, 129:123–137, January 2001. doi: 10.1175/1520-0493(2001)129<0123:ASEKFF>2.0.CO;2.

[45] Rudolf E. Kalman. A New Approach to Linear Filtering and Prediction Problems 1. *ASME*, 82 (Series D):35–45, 1960. doi: 10.1115/1.3662552.

[46] R. R. Karri and V. Babovic. Enhanced predictions of tides and surges through data assimilation. *International Journal of Engineering, Transactions A: Basics*, 30(1):23–29, 2017. doi: 10.5829/idosi.ije.2017.30.01a.04.

[47] Philip D. Kerrison, Michele S. Stanley, Maeve D. Edwards, and Adam D. Black, Kenneth D.and Hughes. The cultivation of European kelp for bioenergy: Site and species selection. *Biomass and Bioenergy*, 80:229–242, September 2015. doi: https://doi.org/10.1016/j.biombioe.2015.04.035.

[48] Ewout Kieckens. European seaweed farms are the future. https://innovationorigins.com/en/european-seaweed-farms-are-the-future/, March 2021. Online accessed on 2021-05-17.

[49] Job Klijnstra, Xiaolong Zhang, Sjoerd van der putten, and Christine Röckmann. *Technical Risks of Offshore Structures*, pages 115–127. April 2017. ISBN 978-3-319-51157-3. doi: 10.1007/978-3-319-51159-7_5.

[50] Eric J. Kostelich, Yang Kuang, Joshua M. McDaniel, Nina Z. Moore, Nikolay L. Martirosyan, and Mark C. Preul. Accurate state estimation from uncertain data and models: an application of data assimilation to mathematical models of human brain tumors, 2011.

[51] Denis Lacroix and Sylvain Pioch. The multi-use in wind farm projects: More conflicts or a win-win opportunity? *Aquatic Living Resources*, 24, 04 2011. doi: 10.1051/alr/2011135.

[52] Arlene Laing and Jenni-Louise Evans. *Introduction to Tropical Meteorology 2nd Edition A Comprehensive Online & Print Textbook*. New York, 2011. http://www.chanthaburi.buu.ac.th/~wirote/met/tropical/textbook_2nd_edition/print_9.htm.

[53] John M. Lewis, S. Lakshmivarahan, and S. Dhall. *Dynamic Data Assimilation, A least Squares Approach*. Cambridge University Press, New York, 2006. ISBN 978-0-521-85155-8. doi: 10.1017/CBO9780511526480.

[54] Xavier Litrico and Vincent Fromion. *Modeling and control of hydrosystems*. 2009. ISBN 9781848826236. doi: 10.1007/978-1-84882-624-3.

[55] Edward Melger. Delft3D FM Suite - Key advantages. Delft3D User Days, 2019.

[56] Baptiste Mourre, Pierre De Mey, Florent Lyard, and Christian Le Provost. Assimilation of sea level data over continental shelves: an ensemble method for the exploration of model errors due to uncertainties in bathymetry. *Dynamics of Atmospheres and Oceans*, 38(2):93–121, 2004. doi: https://doi.org/10.1016/j.dynatmoce.2004.09.001.

[57] Mette Nielsen, Benni Winding Hansen, and Bent Vismann. Feeding traits of the European flat oyster, Ostrea edulis, and the invasive Pacific oyster, Crassostrea gigas. *Marine Biology*, 164: 1–10, 2017. doi: 10.1007/s00227-016-3041-5.

[58] Nadia Pinardi and John Woods. *Ocean Forecasting: Conceptual Basis and Applications*. Springer-Verlag Berlin Heidelberg, New York, 2002. ISBN 978-3-540-67964-6. doi: 10.1007/978-3-662-22648-3.

[59] Nadia Pinardi, L. Cavaleri, Giovanni Coppini, Pierre De Mey-Frémaux, Claudia Fratianni, J. Huthnance, Pierre Lermusiaux, A. Navarra, R. Preller, and Stefano Tibaldi. From weather to ocean predictions: An historical viewpoint. *Journal of Marine Research*, 75:103–159, May 2017. doi: 10.1357/002224017821836789.

[60] Stéphanie Ponsar. *Model state and parameter estimation by assimilation of temperature profiles with an ensemble Kalman filter in North Sea applications*. PhD thesis, Université Catholique de Louvain, 2012.

[61] Stéphanie Ponsar and Patrick Luyten. Data assimilation with the EnKF in a 1-D numerical model of a North Sea station. *Ocean Dynamics*, 59(6):983–996, 2009. ISSN 16167341. doi: 10.1007/s10236-009-0224-3.

[62] Stéphanie Ponsar, Patrick Luyten, and Valérie Dulière. Data assimilation with the ensemble Kalman filter in a numerical model of the North Sea. *Ocean Dynamics*, 66(8):955–971, 2016. ISSN 16167228. doi: 10.1007/s10236-016-0968-5.

[63] Stéphanie Ponsar, Patrick Luyten, and Jose Ozer. Combined model state and parameter estimation with an ensemble kalman filter in a north sea station 1-d numerical model. *Ocean Dynamics*, 61:1869–1886, 11 2011. doi: 10.1007/s10236-011-0477-5.

[64] Joanna Przedrzymirska, Jacek Zaucha, Daniel Depellgrin, Rhona Fairgrieve, Andronikos Kafas, Helena Calado, Marta Vergílio, Mario Caña Varona, Marija Lazic, Angela Schultz-Zehden, Eva Papaioannou, Martina Bocci, Rianne Läkamp, Ioannis Giannelos, Aneta Kovacheva, and Bela Buck. Multi-use of the sea: from research to practice. *SHS Web of Conferences*, 58:1–9, January 2018. doi: 10.1051/shsconf/20185801025.

[65] Lewis F. Richardson. Weather prediction by numerical process. 1922. doi: https://doi.org/10.1002/qj.49704820311.

[66] Leo C. van Rijn. *Principles of fluid flow and surface waves in rivers, estuaries, seas, and oceans*. Aqua Publications, 2nd edition, 1994. ISBN 9080035610.

[67] National Ocean Service. Ioos: Observing for the future. https://celebrating200years.noaa.gov/visions/ioos/obs_system.html, 2007. Online accessed on 2021-04-29.

[68] S. Smolders, Patrick Meire, Stijn Temmerman, Francesco Cozzoli, S. Ides, and Yves Plancke. A 2dh hydrodynamic model of the scheldt estuary in 1955 to assess the ecological past of the estuary. 01 2013.

[69] Ojasvin Sood. Monte carlo simulation in r with focus on option pricing. https://towardsdatascience.com/monte-carlo-simulation-in-r-with-focus-on-financial-data-ad43e2a4aedf, 2019. Online accessed on 2021-05-12.

[70] Wikipedia the free encyclopedia. Hydrostatics. https://en.wikipedia.org/wiki/Hydrostatics, 2021. Online accessed on 2021-08-05.

[71] M. Tonani, P. Sykes, R. R. King, N. McConnell, A.-C. Péquignet, E. O'Dea, J. A. Graham, J. Polton, and J. Siddorn. The impact of a new high-resolution ocean model on the Met Office North-West European Shelf forecasting system. *Ocean Science*, 15(4):1133–1158, 2019. doi: 10.5194/os-15-1133-2019.

[72] Marina Tonani, Magdalena Balmaseda, Laurent Bertino, Ed Blockley, Gary Brassington, Fraser Davidson, Yann Drillet, Pat Hogan, Tsurane Kuragano, Tong Lee, Avichal Mehra, Francis Paranathara, Clemente A.S. Tanajura, and Hui Wang. Status and future of global and regional ocean prediction systems. *Journal of Operational Oceanography*, 8(sup2):s201–s220, 2015. doi: 10.1080/1755876X.2015.1049892.

[73] Harrie van den Akker and Rob Mudde. *Fysische Transportverschijnselen: denken in balansen*, chapter 5.8: De algemene bewegingsvergelijkingen, pages 318–332. Delft Academic Press / VSSD, 4th edition, August 2014.

[74] Johan van der Molen, Piet Ruardij, Karen Mooney, Philip Kerrison, Nessa E. O. Connor, Emma Gorman, Klaas Timmermans, Serena Wright, Maeve Kelly, Adam D. Hughes, and Elisa Capuzzo. Modelling potential production of macroalgae farms in UK and Dutch coastal waters. *Biogeosciences*, 15:1123–1147, 2018. doi: https://doi.org/10.5194/bg-15-1123-2018.

[75] G.J. van der Want. Risk mitigation multi-use offshore wind farms. https://www.topsectorenergie.nl/sites/default/files/uploads/20210329_RAP_Risk_Mitigation_Multi-Use_Offshore_Wind_Farms_MARIN_F_compl..pdf, March 2021. Online accessed on 2021-07-08.

[76] Rijkswaterstaat: Ministerie van Infrastructuur en Waterstaat. MATROOS. https://noos.matroos.rws.nl/, 2021. Online accessed on 2021-8-24.

[77] Sonja van Leeuwen, Paul Tett, David Mills, and Johan van der Molen. Stratified and nonstratified areas in the North Sea: Long-term variability and biological and policy implications. *Journal of Geophysical Research : Oceans*, 120:4670–4686, 2015. doi: 10.1002/2014JC010485.

[78] Nils van Velzen. COSTA a problem solving environment for data assimilation. January 2006.

[79] Nils van Velzen, Muhammad Umer Altaf, and Martin Verlaan. OpenDA-NEMO framework for ocean data assimilation. *Ocean Dynamics*, 66, 2016. doi: 10.1007/s10236-016-0945-z.

[80] Martin Verlaan, Annette Zijderveld, Hans De Vries, and Jan Kroos. Operational storm surge forecasting in the Netherlands: developments in the last decade. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 363:1441–53, 07 2005. doi: 10.1098/rsta.2005.1578.

[81] Peng Wang, Mei Yang, Yong Peng, Jiancheng Zhu, Rusheng Ju, and Quanjun Yin. Sensor control in anti-submarine warfare—a digital twin and random finite sets based approach. *Entropy*, 21:767, August 2019. doi: 10.3390/e21080767.

[82] Albrecht H. Weerts, Ghada Y. El Serafy, Stef Hummel, Juzer Dhondia, and Herman Gerritsen. Application of generic data assimilation tools (DATools) for flood forecasting purposes. *Computers and Geosciences*, 36:453–463, 2010. ISSN 00983004. doi: 10.1016/j.cageo.2009.07.009.

[83] The free encyclopedia Wikipedia. ENIAC. https://en.wikipedia.org/wiki/ENIAC. Online accessed on 2021-07-12.

[84] Patty Zervaas. Multi-use zeeboerderij. https://www.yourwell.nl/voordelen-van-zeewier/, 2021. Online accessed on 2021-07-01.

[85] Firmijn Zijl and Julien Groenenboom. Development of a sixth generation model for the NW European Shelf (DCSM-FM 100m). Technical report, Deltares, 2019.

[86] Firmijn Zijl, Martin Verlaan, and Herman Gerritsen. Improved water-level forecasting for the Northwest European Shelf and North Sea through direct modelling of tide, surge and non-linear interaction. *Ocean Dynamics*, 63, July 2013. doi: 10.1007/s10236-013-0624-2.

[87] Firmijn Zijl, Julius Sumihar, and Martin Verlaan. Application of data assimilation for improved operational water level forecasting on the northwest European shelf and North Sea. *Ocean Dynamics*, 65, 2015. doi: 10.1007/s10236-015-0898-7.

[88] Firmijn Zijl, Jelmer Veenstra, and Julien Groenenboom. The 3D Dutch Continental Shelf Model - Flexible Mesh (3D DCSM-FM). Technical report, Deltares, 2018.